Internal Note No. 68-FM-32

Blc L10

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

MSC INTERNAL NOTE NO. 68-FM-32

February 26, 1968

N.I.

RTCC REQUIREMENTS FOR MISSION G: FIRST-GUESS LOGIC FOR THE TLI PROCESSOR

By Francis Johnson, Jr. and Thomas J. Linbeck
Lunar Mission Analysis Branch

MISSION PLANNING AND ANALYSIS DIVISION

MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

(NASA-TM-X-69371) RTCC MISSION REQUIREMENTS FOR MISSION G: FIRST-GUESS LOGIC FOR THE TLI PROCESSOR (NASA)
107 p

N74-70631

Unclas
00/99 16178

MSC INTERNAL NOTE NO. 68-FM-32

PROJECT APOLLO

RTCC REQUIREMENTS FOR MISSION G: FIRST-GUESS LOGIC FOR THE TLI PROCESSOR

By Francis Johnson, Jr. and Thomas J. Linbeck
Lunar Mission Analysis Branch

February 26, 1968

MISSION PLANNING AND ANALYSIS DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

Approved: M P Frank
M. P. Frank III, Chief
Lunar Mission Analysis Branch

Approved: John P Mayer
John P. Mayer, Chief
Mission Planning and Analysis Division

FOREWORD

First guesses for the RTCC TLI processor are obtained from the use of two subroutines, CIST and UPDATE. CIST defines the coplanar injection into a simulated trajectory from an earth parking orbit. UPDATE modifies the coplanar TLI targeting elements output by CIST to compensate for a dispersion in the time of the parking orbit state vector. This document describes how these two subroutines work, and how their satellite subroutines work.

This document is, in effect, a resume of literature distributed during the past year and a half describing subroutines CIST and UPDATE and their use. This literature consists of numbered memorandums and informal handouts. The basic logics of CIST and UPDATE as described herein, have not been changed since these subroutines were first written.

I/F & CONTROL LOGIC DEF'D

IN TLI SUPERVISOR MSC IN 68-FM-28

CONTENTS

Section		Page

Section Page

Section		Page

Section | Page

RTCC REQUIREMENTS FOR MISSION G: FIRST-GUESS

LOGIC FOR THE TLI PROCESSOR

By Francis Johnson, Jr., and Thomas J. Linbeck

1.0 SUBROUTINE CIST

1.1 Purpose

CIST defines the coplanar injection into a simulated trajectory from a specific earth parking orbit. In doing this, CIST defines the TLI targeting elements, state vectors at the beginning and end of the coplanar TLI thrust maneuver, the perigee state vector of the outgoing trajectory, and the time of a parking orbit state vector, the position and velocity variables of which are input.

The basic independent variable of the primary iteration loop in CIST can be regarded as being either the time of pericynthion (TOPCY) or the argument (A2M) in the moon's orbit plane (MOP) of the node between the MOP and vehicle's orbit plane (VOP). Both TOPCY and A2M are iteratively adjusted so as to drive the difference between available flight time and required flight time to zero. Available flight time is the difference between TOPCY and the time of trajectory perigee. Required flight time is that of the simulated trajectory as defined by empirical equations.

The simulated trajectory in the present version of CIST is the familiar S-shaped trajectory having pericynthion on the back side of the moon. The simulation of this type of trajectory is described in reference 1. The following is a list of the twelve variables completely defining the perigee and pericynthion state vectors of the simulated trajectory. In reference 1 the dependence and independence of these variables are defined in terms of individual trajectory calculation. As indicated below, the dependence and independence of these variables in terms of CIST usage is different.

Pericynthion variables of simulated trajectory		
Independent in reference 1	TOPCY	Dependent, iteratively defined, in CIST
	XPC YPC RPC	Independent, input to, CIST
Dependent in reference 1	AZPC WPC	Not used or defined in present version of CIST

Perigee variables of simulated trajectory		
Independent in reference 1	FIVTL RPG	Dependent, iteratively defined, in CIST
Dependent in reference 1	FT WPG XPG YPG	

The simulation of the coplanar TLI thrust maneuver used in CIST is described in reference 2. This simulation can accomodate different parking orbit altitudes, thrust-to-weight ratios, any trajectory energy, and is more accurate than simulations using multiple sets of polynomials.

1.2 System of Naming Addresses of Variables in CIST

There is a large number of variables involved in CIST logic. Experience over the past few years has shown that in the processes of debugging and making program modifications, it is very easy for the programer to flounder in this maze of variables unless he adheres to a consistent system of naming the addresses of these variables. The following system, used in the present version of CIST, has been found very convenient to work with, and it is felt to be very adaptable to any future programing modifications.

The general logic of CIST performs an iterative juggling act of satisfying necessary time-geometry relationships between eight different events. These events are listed below with their corresponding numerical identifications which will appear in the address of any variable describing the respective event:

1. Pickup state in earth parking orbit (EPO).

2. Node between the VOP and MOP in the vicinity of translunar injection (TLI).

3. The target vector $\hat{M}$ for TLI guidance.

4. Perigee of the translunar trajectory.

5. The impulsive (Δr, $\Delta\gamma$, ΔV) simulation of the TLI maneuver.

6. Pericynthion nadir, the negative unit position vector of the moon at the time of trajectory pericynthion.

7. Cutoff of the actual TLI thrust maneuver.

8. Initiation of the actual TLI thrust maneuver.

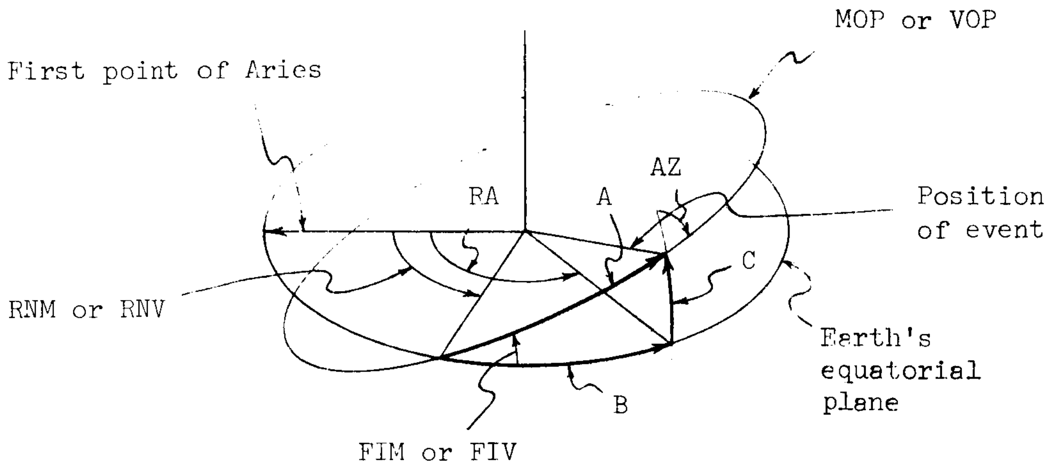
For instance, any address containing the number 4 will describe the trajectory perigee. The numerical order of the above events does not imply any chronological order.

The positions of these eight events occur in two planes, the VOP and the MOP. The inertial orientations of these two planes are defined by the right ascensions of their ascending nodes on the earth's equator and their inclinations to the earth's equator. Addresses of these ascending node right ascensions are RN, and addresses of these inclinations are FI, with the addition of either of the letters V and M to denote whether the plane is of the vehicle's or moon's orbit. FIV and FIM will always be positive, between 0 and 90°. RNV and RNM will always be between $-\pi$ and π.

Five different angles are used to describe an event in the appropriate orbit plane (MOP or VOP). The following alphabetic system is used to identify these different angles:

A	Argument measured in orbit plane from its ascending node on the earth's equator
B	Longitude measured in the earth's equatorial plane from the orbit's ascending node
C	Declination relative to the earth's equator
RA	Conventional right ascension
AZ	Conventional azimuth

The following figure illustrates the use of these five angles in describing an event or position in either the VOP or the MOP.



It can be seen that the RA and C of a position are independent of which orbit plane the position lies in. However, the A, B, and AZ of a position must relate to either one of the two orbit planes.

Of the eight different events described on the preceeding page, only one of them, pericynthion nadir (6), always occurs in the MOP and never occurs in the VOP. Consequently, the addresses A6, B6, and AZ6 will always represent angles describing pericynthion nadir in the MOP.

With one exception, all of the other events will always lie in the VOP. Consequently, there does not have to be any confusion as to which orbit plane the angles A, B, and AZ of these events relate. The one exception is the node (2) between the VOP and the MOP, which by definition will always lie in both planes. To eliminate the ambiguities of the notations A2, B2, and AZ2, the letter M or V is added to denote whether the orbit plane concerned is the moon's or the vehicle's.

Additional alphabetic notations of variables describing an event are as follows:

T   Time, relative to base time in hours

R   Radius in nautical miles

FLO Geographic longitude

V Velocity in international feet per second

All angles are expressed in radians. The address of an angle expressed in degrees for printout purposes is the same as its address as expressed in radians with the addition of the letter D.

1.3 Input To and Output From CIST, the SIMUL Common Block

With the sole exception of the single variable (RAGBT) in the calling argument of CIST, all input to and output from CIST occurs through a common block (SIMUL) having 100 locations. RAGBT is the right ascension of Greenwich in radians at the base hour PRE(3).

The SIMUL block is in common with the program calling CIST, CIST, most of the subroutines of CIST, and UPDATE. There are four arrays in SIMUL. All variables in SIMUL are real.

COMMON/SIMUL/PRE(30), XMS(25), TAR(25), TRAJ(20)

1.3.1 PRE array.- The PRE array contains all of the input to CIST. All of this array is not now used in the present version of CIST. Space is made available for the additional input to a more sophisticated version of CIST wherein the simulated trajectory can be specified with greater flexibility. (In the present version, only the position of pericynthion can be specified.)

Location in SIMUL block	Location in PRE array	MNEMONIC	Description
1	1		Year during which the time of the input orbit state vector is to be defined.
2	2		Number of the day in the year in which the midpoint (BH) occurs of the 24-hour period within which the input orbit state vector is to be defined.
3	3	BH	G.m.t. midpoint of the 24-hour period during which the time of the input orbit state vector is to be defined, in hours.
4	4	C1D	Declination of the input orbit state vector, in degrees.

Location in SIMUL block	Location in PRE array	MNEMONIC	Description
5	5	FLO1D	Geographic longitude of the input orbit state vector, in degrees.
6	6	AZ1D	Inertial azimuth of the input orbit state vector, in degrees.
7	7	R1	Radius of the input orbit state vector, in nautical miles.
10	10	ORBNUM	Number of the inertial orbit revolution, from the input orbit state vector, during which TLI is to begin. ORBNUM = 0.0 allows negative orbit coast times which must be considered when the input orbit state vector is an approximation to the beginning of the TLI maneuver.
11	11	FIPOA	Instruction index indicating the "window" in which TLI is to occur. FIPOA = 1.0, the Pacific window ($AZ2V \leq \pi/2$) FIPOA = 2.0, the Atlantic window ($AZ2V > \pi/2$)
12	12	FIPERT	Instruction index indicating which perturbations are to be considered in calculating the simulated trajectory. FIPERT = 0.0, No perturbations considered = 1.0, Only earth oblateness considered = 2.0, Only solar gravitation considered = 3.0, Both earth oblateness and solar gravitation considered

Location in SIMUL block	Location in PRE array	MNEMONIC	Description
13	13	XPC	Longitude of pericynthion of the simulated trajectory, measured in a moon-centered MOP coordinate system from the extension of the earth-moon axis on the back side of the moon, in degrees.
14	14	YPC	Declination of pericynthion of the simulated trajectory, in a moon-centered MOP coordinate system, in degrees.
15	15	RPC	Radius of pericynthion of the simulated trajectory, relative to the center of the moon, in nautical miles.
16	16		Maximum permissible number of iterations in CIST. A very safe number is 50.0. The average number of iterations for convergence is observed to be about 5.
17	17		Convergence tolerance of time of pericynthion in hours. This is the maximum permissible magnitude of the difference between the time of pericynthion defined by ephemeris call, and that defined by required flight time. A reasonable value is 0.001.
29	29	TTW	Thrust-to-weight ratio at the beginning of the TLI maneuver.

1.3.2 XMS array.- The XMS array contains all of the variables describing the positions of the sun and pericynthion nadir, and the position and motion of the moon at the time of pericynthion of the simulated trajectory. All of the variables in the XMS array are defined by subroutine PERCYN.

Location in SIMUL block	Location in XMS array	MNEMONIC	Description
31	1	EMR	Earth-moon radius in nautical miles
32	2	EMRDOT	$\dot{EMR}$, first derivative of earth-moon radius with respect to time, in knots.
34	4	FIM	Inclination of the MOP to earth's equatorial plane, in radians.
35	5	RNM	Right ascension of the ascending node of the MOP, in radians. $-\pi < RNM \leq \pi$
36	6	AM	Argument of the moon in the MOP past its ascending node on the earth's equator, in radians. $-\pi < AM \leq \pi$
37	7	RAM	Right ascension of the moon, in radians. $-\pi < RAM \leq \pi$
38	8	DECM	Declination of the moon, in radians.
39	9	A6	Argument of pericynthion nadir in MOP past moon's ascending node on earth's equator, in radians. $-\pi < A6 \leq \pi$
40	10	RA6	Right ascension of pericynthion nadir, in radians. $-\pi < RA6 \leq \pi$
41	11	C6	Declination of pericynthion nadir, in radians.
42	12	B6	Longitude of pericynthion, measured in earth's equatorial plane from ascending node of MOP, in radians. $-\pi < B6 \leq \pi$

Location in SIMUL block	Location in XMS array	MNEMONIC	Description
43	13	AZ6	Azimuth of MOP at pericynthion nadir, in radians.
44	14	WM	Angular velocity of the moon, in radians/hr.
45	15	XHM	Cartesian components of the angular momentum vector of the moon in e.r.^{2}/hr, relative to the earth's center.
46	16	YHM	
47	17	XHM	
51	21	SMOPL	Longitude of the sun in an earth-centered MOP coordinate system, measured counter-clockwise from the position of the moon, in radians.
			$-\pi < \text{SMOPL} \leq \pi$
52	22	SMOPD	Declination of the sun in an earth-centered MOP coordinate system, in radians.
53	23	AS	Argument of the sun in the ecliptic past its ascending node on the earth's equatorial plane, in radians.
			$-\pi < \text{AS} \leq \pi$
54	24	RAS	Right ascension of the sun, in radians.
			$-\pi < \text{RAS} \leq \pi$
55	25	DS	Declination of the sun, in radians.

1.3.3 TAR array.- The TAR array contains the more significant output of CIST, consisting of the coplanar TLI targeting elements and all of the variables used by subroutine UPDATE to modify these elements. TAR locations 21 through 25 are for the TLI targeting elements modified by UPDATE.

Location in SIMUL block	Location in TAR array	MNEMONIC	Description
56	1	COI	Convergence index, indicating whether CIST was successful in achieving convergence. COI = 0.0, Satisfactory convergence acheived = 1.0, Maximum number of iterations performed without achieving convergence = 2.0, Ephemeris trouble, no convergence = 3.0, Coplanar solution not attainable; Non-zero δ necessary. No convergence.
57	2	T1	Time of the input orbit state vector, relative to base time [PRE(3)], in hours.
58	3	WV	Angular velocity of the vehicle in earth parking orbit, in rad/hr.
59	4	WE	Angular velocity of the earth's rotation, in rad/hr.
60	5	FIV	Inclination of the earth parking orbit to the earth's equator, in radians.
61	6	FIVTL	Inclination of the simulated trajectory at perigee to the MOP, in radians. FIVTL is defined as negative if the trajectory is going below the MOP.
63	8	C3	Declination of the $\hat{M}$ unit TLI target vector defined by CIST for coplanar TLI, in radians.
64	9	RA3	Right ascension of the $\hat{M}$ unit TLI target vector defined by CIST for coplanar TLI, in radians.

Location in SIMUL block	Location in TAR array	MNEMONIC	Description
65	10	S	The angle σ, the radius of the hypersurface representing all possible perigee positions, in radians.
66	11	W	Energy of the simulated translunar trajectory at perigee, in (international ft/sec^2) $W = \frac{V^2}{2} - \frac{\mu}{r}$
67	12	XMH	Cartesian components of the unit $\hat{M}$ TLI targeting vector defined by CIST for coplanar TLI.
68	13	YMH	
69	14	ZMH	
70	15	CTIO	Coast time from the input parking orbit state vector without time dispersion to the beginning of the coplanar TLI thrust maneuver, in hours. [CTIO can be negative in cases where PRE(10) is 0.0.]
71	16	RNV	Right ascension of the ascending node of the parking orbit on the earth's equator, in radians.
74	19	A1	Argument of the input parking orbit state vector past its ascending node on the earth's equator, in radians.
76	21	DEL	The angle δ, the latitude in radians, of the unit $\hat{M}$ TLI target vector after it has been modified by subroutine UPDATE, relative to the plane of the parking orbit with time dispersion, in radians.
78	23	XUMH	Cartesian components of the unit $\hat{M}$ TLI target vector after it has been modified by subroutine UPDATE to compensate for a time dispersion of the input parking orbit state vector.
79	24	YUMH	
80	25	ZUMH	

1.3.4 TRAJ array.- The TRAJ array contains variables describing the simulation of the coplanar TLI thrust maneuver. All of these variables are defined in subroutine TLIMP.

Location in SIMUL block	Location in TRAJ array	MNEMONIC	Description
81	1	ETA	True anomaly, in radians, of the cutoff of coplanar TLI on the translunar trajectory.
82	2	APS	Angle measured in the parking orbit-trajectory plane from the beginning of the coplanar TLI thrust maneuver to trajectory perigee. (Stands for Alfa Plus Sigma, where α and σ are the established TLI angles.) In radians.
83	3	G7D	Flight-path angle, in degrees, of the cutoff of the coplanar TLI thrust maneuver, on the translunar trajectory.
84	4	DV	The characteristic velocity of the coplanar TLI thrust maneuver, in international fps.
85	5	FTOCO	Flight time of cutoff of coplanar TLI on the translunar trajectory past perigee, in hours.
86	6	R7	Radius of cutoff of the coplanar TLI thrust maneuver, in nautical miles.
87	7	E	Eccentricity of the translunar trajectory at perigee.
88	8	R4	Perigee radius — Conic elements of the translunar trajectory at perigee in nautical miles
89	9	P	Semilatus rectum — Conic elements of the translunar trajectory at perigee in nautical miles
90	10	A	Semimajor axis — Conic elements of the translunar trajectory at perigee in nautical miles
91	11	B	Semiminor axis — Conic elements of the translunar trajectory at perigee in nautical miles
92	12	COEF	Coefficient of flight time equation, in hr/n. mi.

1.4 General Logic of CIST, the Primary Iteration Loop

The complete logic of CIST is best described in two phases: First, the general logic, the logic of the primary iteration loop. Second, specialized logics which are designed to deal with special situations which the aforementioned general logic would be incapable of dealing with. The general logic is described in this section. These specialized logics take the form of discrete blocks of consecutive statements in the listing of CIST. These specialized logic blocks are described in detail in section 1.5.

There are several blocks of consecutive statements in the CIST listing which have no bearing on the iteration process. The sole purpose of these statement blocks is to provide printout. These statement blocks can be eliminated from CIST without impairing its accuracy or reliability. These printout statement blocks are identified as such in the listing of CIST.

First, there is a multitude of EQUIVALENCE statements which enable coding with descriptive alpha-numeric addresses instead of subscripted variables.

```
SUBROUTINE CIST (RAGBT)
COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
EQUIVALENCE (PRE(3),BH)
EQUIVALENCE (PRE(4),C1D)
EQUIVALENCE (PRE(5),FL01D)
EQUIVALENCE (PRE(6),AZ1D)
EQUIVALENCE (PRE(7),R1)
EQUIVALENCE (PRE(10),ORBNUM)
EQUIVALENCE (PRE(11),FIPOA)
EQUIVALENCE (PRE(12),FIPERT)
EQUIVALENCE (PRE(13),XPC)
EQUIVALENCE (PRE(14),YPC)
EQUIVALENCE (PRE(15),RPC)
EQUIVALENCE (PRE(29),TTW)
EQUIVALENCE (XMS(4),FIM)
EQUIVALENCE (XMS(5),RNM)
EQUIVALENCE (XMS(9),A6)
EQUIVALENCE (XMS(10),RA6)
EQUIVALENCE (XMS(11),C6)
EQUIVALENCE (XMS(12),B6)
EQUIVALENCE (XMS(13),AZ6)
EQUIVALENCE (XMS(14),WM)
EQUIVALENCE (TAR(1),COI)
EQUIVALENCE (TAR(2),T1)
```

```
EQUIVALENCE (TAR(3),WV)
EQUIVALENCE (TAR(4),WE)
EQUIVALENCE (TAR(5),FIV)
EQUIVALENCE (TAR(6),FIVTL)
EQUIVALENCE (TAR(7),AZ3)
EQUIVALENCE (TAR(8),C3)
EQUIVALENCE (TAR(9),RA3)
EQUIVALENCE (TAR(10),S)
EQUIVALENCE (TAR(11),W)
EQUIVALENCE (TAR(12),XMH)
EQUIVALENCE (TAR(13),YMH)
EQUIVALENCE (TAR(14),ZMH)
EQUIVALENCE (TAR(15),CTIO)
EQUIVALENCE (TAR(16),RNV)
EQUIVALENCE (TAR(17),A4)
EQUIVALENCE (TAR(18),CTTPG)
EQUIVALENCE (TAR(19),A1)
EQUIVALENCE (TRAJ(1),ETA)
EQUIVALENCE (TRAJ(2),APS)
EQUIVALENCE (TRAJ(3),G7D)
EQUIVALENCE (TRAJ(4),DV)
EQUIVALENCE (TRAJ(5),FTOCO)
EQUIVALENCE (TRAJ(6),R7)
EQUIVALENCE (TRAJ(8),R4)
```

The constants DPR, PI, U, and WE are first defined. Input values of declination, longitude, and azimuth of the parking orbit pickup state are converted to radians. Input indices are converted to fixed point values. Then the following calculations occur: the angular velocity (WV) of the vehicle in parking orbit, the orbit period, the angles B1 and A1, FIV, and the right ascension of the ascending node of the VOP for insertion at base time (RNVBT). The 12 statements following these calculations are for initial printout.

```
DPR=57.29578
PI=3.1415927
U=.2315700400E+13
WE=.262516142
C1=C1D/DPR
FL01=FL01D/DPR
AZ1=AZ1D/DPR
IORBN=ORBNUM
IPOA=FIPOA+0.1
IPERT=FIPERT+0.1
WV=SQRT(19.9094165/(R1/3443.93359)**3)
```

```
      ORBPER=2.0*PI/WV
      B1=ATAN2(SIN(C1)*SIN(AZ1),COS(AZ1))
      A1=ATAN2(SIN(C1),COS(C1)*COS(AZ1))
      FIV=ABS(ATAN(SIN(C1)/(COS(C1)*SIN(B1))))
      RNVBT=RAGBT+FL01-B1
      CALL HELP (RNVBT)
C
C                                             THE NEXT 12 STATEMENTS
C                                             ARE FOR INITIAL PRINTOUT
      IY=PRE(1)+0.1
      ID=PRE(2)+0.1
      FIVD=FIV*DPR
      RAGBTD=RAGBT*DPR
      A1D=A1*DPR
      RNVBTD=RNVBT*DPR
      B1D=B1*DPR
      WRITE (6,913) IY,FL01D,R1,XPC,ID,C1D,IORBN,YPC,
     1 BH,AZ1D,FIVD,RPC,RAGBTD,A1D,ORBPER
      IF (IPOA.EQ.2) WRITE (6,914) RNVBTD,B1D,IPERT
      IF (IPOA.EQ.1) WRITE (6,915) RNVBTD,B1D,IPERT
      WRITE(6,900)
C
C
```

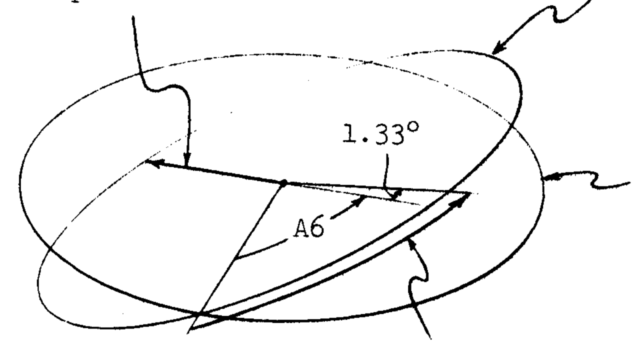
A first guess is made of the earliest possible time of pericynthion (TOPCY) relative to base time. Assumptions are made of the earliest possible time of insertion, shortest coast time in orbit, and shortest flight time. The convergence index (COI) is defined as 0.0, signifying that all is well. PERCYN is called thus defining all values in the XMS array for the time TOPCY. If ephemeris trouble is encountered in PERCYN, COI is set equal to 2.0. If this condition is detected, control is returned to the program which called CIST.

```
      TOPCY=ORBPER*(ORBNUM-1.0)+40.0
      COI=0.0
      CALL PERCYN (TOPCY)
      IF(COI.NE.2.0) GO TO 490
      RETURN
```

W is the energy of the simulated trajectory at perigee and RQDFT is the (required) flight time of the simulated trajectory from perigee to pericynthion. Values of W and RQDFT are defined by calling subroutines ENERGY and FLYTYM. The empirical equations for W and RQDFT in these subroutines have as variables: FIVTL, C4, and R4. At this point in the logic, these variables have not been defined. Reasonable values of these variables are defined before subroutines ENERGY and FLYTYM are called. It should be remembered that at this point in the logic, the accuracy of these variables is not critical. Accurate values are defined in the primary iteration loop.

```
490   FIVTL=0.0
      C4=0.0
      R4=R1+12.0
      CALL ENERGY (IPERT,C4)
      CALL FLYTYM (IPERT,RQDFT)
```

Assuming a BETA of -1.33°, an "earliest possible" position of the node between the MOP and VOP in the vicinity of TLI is defined relative to the already defined "earliest possible" pericynthion nadir, the argument of the latter being A6. The position of this node is defined by the argument A2M in the MOP, as shown in the following figure. The iteration count (NUMIT) and the indices and test values for specialized logics are initialized.

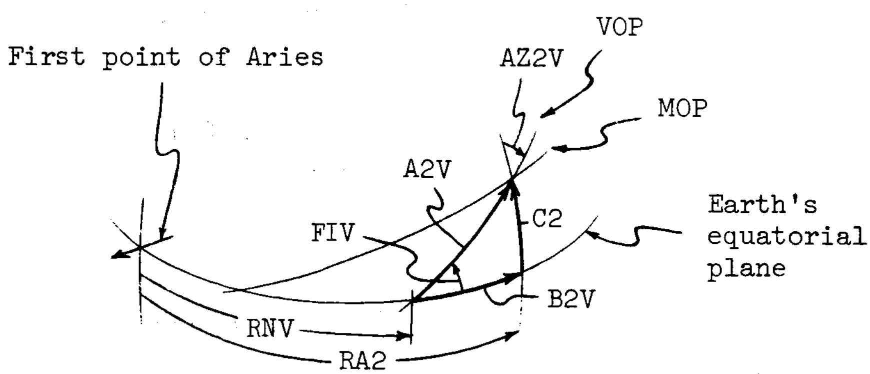



```
      A2M=A6+1.33/DPR
      CALL HELP (A2M)
      PT1=0.0
      NUMIT=0
      IEX=0
      IA10=1
      IOS=1
```

The primary iteration loop begins at statement 500, at which the iteration count is incremented. GEOARG is called, thus defining AZ2M, B2M, C2, and RA2. A test is then made to be sure that the node on the MOP can be "reached" by the VOP. If it cannot be reached, a specialized logic block of 25 statements must be resorted to. If it can be reached, GO TO 650. The inaccessible node logic block is described in section 1.5.1.

```
C
C                                                   THIS IS THE BEGINNING
C                                                   OF THE ITERATION LOOP
  500  NUMIT=NUMIT+1
       CALL GEOARG (FIM,A2M,RNM,AZ2M,B2M,C2,RA2)
       IF(ABS(C2).LT.FIV) GO TO 650
C
C                                                   THE NEXT 25 STATEMENTS ARE
C                                                   INACCESSIBLE NODE LOGIC
                                                    (See listing in section 1.5.1.)
```

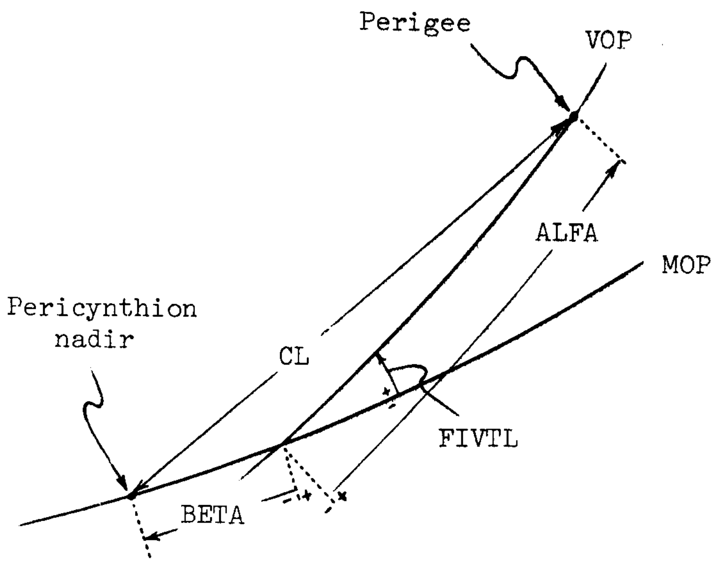
At statement 650, subroutine GEOLAT is called, thus defining the angles describing the inertial orientation of the VOP and the position of the node in the VOP such that AZ2V is in the proper quadrant as prescribed by the index IPOA. When IPOA is 1, AZ2V is in the first quadrant, resulting in TLI out of the "Pacific" window. When IPOA is 2, AZ2V is in the second quadrant, resulting in TLI out of the "Atlantic" window. The geometry involved is illustrated in the following figure.



FIVTL is calculated and the subroutines SUBB, SUBCL, and TLIMP are called. Calling subroutines SUBB and SUBCL defines the angles BETA and CL.

```
550  CALL GEOLAT (FIV,C2,RA2,IPOA,A2V,B2V,AZ2V,RNV)
     FIVTL=AZ2M-AZ2V
     CALL SUBB (IPERT,BETA,AFNTMH)
     CALL SUBCL (IPERT,CL)
     CALL TLIMP
```

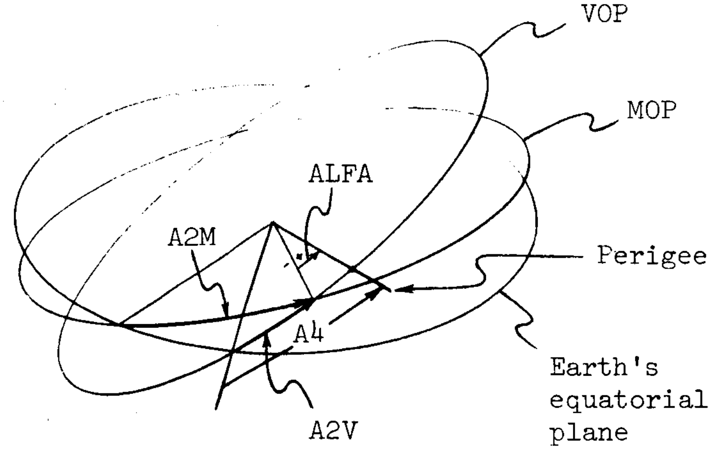
These angles are illustrated in the next figure.



Once the values of CL, BETA, and FIVTL are calculated, the value of ALFA is calculated. This latter calculation involves the dummy variable Z in the present coding. After ALFA is calculated, the value of A4 is calculated.

```
Z=ATAN(SIN(BETA)*COS(FIVTL)/COS(BETA))
ALFA=(COS(BETA)/(COS(Z)*COS(CL)))**2-1.0
IF(ALFA.LT.0.0) ALFA=0.0
ALFA=ATAN(SQRT(ALFA))+Z
A4=A2V+ALFA
CALL HELP (A4)
CALL GEOARG (FIV,A4,RNV,AZ4,B4,C4,RA4)
```

The geometry involved is illustrated in the following figure.



After A4 is defined, the other angles (AZ4, B4, C4, and RA4) describing perigee are defined by calling GEOARG. C4 will be used in subroutine ENERGY in calculating trajectory energy (W).

The coast arc (CARC) in the earth parking orbit, from the input pickup state vector to the beginning of the TLI thrust maneuver, is calculated. The angle APS (stands for Alpha Plus Sigma), which was defined by calling subroutine TLIMP, is used in doing this. CARC is defined by mod 2π according to IORBN, the instruction index of orbit number during which injection occurs.

When IORBN is not equal to zero, CARC is defined between 0 and 2π. When IORBN is zero, CARC is defined between $-\pi$ and π. This latter definition, permitting negative values of CARC, is necessary when the orbit pickup state vector is an approximation to the beginning of TLI.

A test is made of the change in CARC since the last iteration. The value of CARC in the last iteration is PCARC. If the absolute value of the change in CARC is greater than π, a specialized logic block which is designed to deal with the problem of orbit coast time excursions must be resorted to. On the first iteration, this specialized logic is always bypassed by defining PCARC as equal to CARC immediately prior to the test. If no orbit coast time excursion is detected, GO TO 657. The orbit coast time excursion logic is described in section 1.5.2.

```
      CARC=A4-A1-APS
      CALL HELP (CARC)
      IF(CARC.LT.0.0.AND.IORBN.NE.0) CARC=CARC+2.0*PI
      IF(NUMIT.EQ.1) PCARC=CARC
      IF(ABS(CARC-PCARC).LT.PI) GO TO 657
C
C                                   THE NEXT 8 STATEMENTS ARE ORBIT
C                                   COAST TIME EXCURSION LOGIC
```

(See listing in section 1.5.2.)

At statement 657, PCARC is defined as equal to CARC for the test for orbit coast time excursion during the next iteration.

The coast time in orbit (CTIO) from insertion to the beginning of TLI is calculated. When IORBN is 0, negative values (hopefully, very small) of CTIO are allowed. The time of the orbit pickup state vector (T1) relative to base time is calculated as the difference between the immediate right ascension (RNV) of the ascending node of the VOP and that value when insertion occurs at base time (RNVBT), divided by the angular velocity of the earth's rotation (WV). The difference between RNV and RNVBT must first be defined between $-\pi$ and π by calling subroutine HELP in order to constrain T1 to within $\pm$ 12 hours of base time.

A test is then made to see if the change in T1, from its value (PT1) in the previous iteration, is greater than 12 hours. If the change is greater than 12 hours, a specialized logic block is resorted to which is designed to deal with the problem of excursions in T1. This specialized logic is described in a section 1.5.3. If the change in T1 is less than 12 hours, GO TO 680.

```
  657 PCARC=CARC
      IF(IORBN.NE.0) CTIO=ORBPER*(CARC/(2.0*PI)+ORBNUM-1.0)
      IF(IORBN.EQ.0) CTIO=ORBPER*(CARC/(2.0*PI))
      T1=RNV-RNVBT
      CALL HELP (T1)
      T1=T1/WE
      IF(NUMIT.EQ.1) PT1=T1
      IF(ABS(T1-PT1).LT.12.0) GO TO 680
C
C                                   THE NEXT 6 STATEMENTS ARE TIME
C                                   OF ORBIT STATE VECTOR
C                                   EXCURSION LOGIC
```

(See listing in section 1.5.2.)

At statement 680, PT1 is defined as equal to T1 for the excursion test in the next iteration.

A required time of pericynthion (RTOPCY) is calculated based on immediate values of T1, CTIO, and RQDFT. In doing this, it is necessary to take into consideration the coast time (DUMCT) in orbit over the angle APS because CTIO is measured up to the beginning of TLI, whereas RQDFT is measured from perigee. The necessary change (DTOPCY) in TOPCY is then calculated.

```
 680  PT1=T1
      DUMCT=APS/WV
      CTTPG=CTIO+DUMCT
      RTOPCY=T1+CTIO+DUMCT+RQDFT
      DTOPCY=RTOPCY-TOPCY
```

The next block of statements is for the printout of several significant variables which indicate what is happening in the primary iteration loop.

```
C
C                                                THE NEXT  9 STATEMENTS ARE FOR
C                                                ITERATION HISTORY PRINTOUT
C
      C2D=C2*DPR
      AZ2VD=AZ2V*DPR
      FIVTLD=FIVTL*DPR
      BETAD=BETA*DPR
      CLD=CL*DPR
      T8=T1+CTIO
      T4=T8+DUMCT
      T7=T4+FTOCO
      WRITE (6,901) NUMIT,C2D,AZ2VD,FIVTLD,BETAD,CLD,W,RQDFT,
     1 T1,T8,T4,T7,RTOPCY,TOPCY,DTOPCY
C
C
```

A test is made to see if the absolute value of DTOPCY is less than the input tolerance, PRE(17). If |DTOPCY| is less than PRE(17), the iteration process is terminated by going to statement 800.

A test is then made to see if the maximum permissable number of iterations (MAXIT) has been reached. If MAXIT has been reached, the iteration process is terminated by going to statement 700 where a statement to this effect is printed out and the convergence index is set equal to 1.0.

```
      IF(ABS(DTOPCY).LE.PRE(17)) GO TO 800
      MAXIT=PRE(16)+0.1
      IF(NUMIT.GE.MAXIT) GO TO 700
```

If, after making these two tests, the iteration is not terminated, TOPCY is defined as equal to RTOPCY and PERCYN is called, thus defining a new set of variables in the XMS array.

Subroutines ENERGY and FLYTYM are then called, thus defining new values of trajectory energy (W) and required flight time (RQDFT) which are compatible with the new XMS variables.

A new position of the node between the VOP and the MOP is defined by calculating a new value of A2M. This calculation is based upon the approximation that BETA will not change significantly from one iteration to the next. A2M is constrained between $-\pi$ and π by calling subroutine HELP.

The next iteration is begun by going to statement 500.

```
      TOPCY=RTOPCY
      CALL PERCYN (TOPCY)
      IF(COI.NE.2.0) GO TO 590
      RETURN
590   CALL ENERGY (IPERT,C4)
      CALL FLYTYM (IPERT,RQDFT)
      A2M=A6-BETA
      CALL HELP (A2M)
      GO TO 500
700   WRITE (6,902) MAXIT
      COI=1.0
```

At the termination of the iteration process, variables describing TLI targeting and the state vectors of perigee and those at the beginning and end of the TLI thrust maneuver are calculated for external use and immediate printout. These calculations begin at statement 800.

```
800   A3=A2V+AFNTMH
      CALL HELP (A3)
      CALL GEOARG (FIV,A3,RNV,AZ3,B3,C3,RA3)
      S=A4-A3
      CALL HELP (S)
      SD=S*DPR
      XMH=COS(C3)*COS(RA3)
      YMH=COS(C3)*SIN(RA3)
      ZMH=SIN(C3)
```

```
      C3D=C3*DPR
      RA3D=RA3*DPR
      ANMHD=AFNTMH*DPR
      ETAD=ETA*DPR
      ALFAD=(APS-S)*DPR
      WTT=W*2.0
      WTTK=WTT/3280.8399**2
      WTTER=WTT*(3600.0/(6076.11549*3443.93359))**2
      WRITE (6,911) C3D,SD,RA3D,ALFAD,W,ETAD,WTT,ANMHD,WTTK,
     1 DV,WTTER,TTW
      C4D=C4*DPR
      RA4D=RA4*DPR
      AZ4D=AZ4*DPR
      V4=SQRT(2.0*(W+U/R4))
      FLO4=RA4-RAGBT-T4*WE
      CALL HELP (FLO4)
      FLO4D=FLO4*DPR
      A8=A4-APS
      CALL HELP (A8)
      CALL GEOARG (FIV,A8,RNV,AZ8,B8,C8,RA8)
      C8D=C8*DPR
      RA8D=RA8*DPR
      AZ8D=AZ8*DPR
      V8=SQRT(U/R1)
      FLO8=RA8-RAGBT-T8*WE
      CALL HELP (FLO8)
      FLO8D=FLO8*DPR
      A7=A4+ETA
      CALL HELP (A7)
      CALL GEOARG (FIV,A7,RNV,AZ7,B7,C7,RA7)
      C7D=C7*DPR
      RA7D=RA7*DPR
      AZ7D=AZ7*DPR
      V7=SQRT(2.0*(W+U/R7))
      FLO7=RA7-RAGBT-T7*WE
      CALL HELP (FLO7)
      FLO7D=FLO7*DPR
      WRITE (6,912) T8,T4,T7,C8D,C4D,C7D,RA8D,RA4D,RA7D,
     1 AZ8D,AZ4D,AZ7D,FLO8D,FLO4D,FLO7D,R1,R4,R7,V8,V4,V7,G7D
      RETURN
```

All of these calculations are based upon the empirical simulation of optimum coplanar TLI thrust maneuvers described in reference 2. All of the variables of this simulation are defined in subroutine TLIMP.

The $\hat{M}$ defined is the point of tangency of the VOP on the translunar injection tangency surface. This method of defining $\hat{M}$ is described in reference 3.

The following are the format statements used in all CIST printout.

```
900  FORMAT (///18X,2HC2,14X,4HAZ2V,13X,4HIVTL,13X,4HBETA,14X,2HCL,
   1 15X,1HW,15X,5HRQDFT/
   2 18X,2HT1,15X,2HT8,15X,2HT4,15X,2HT7,11X,10HRQD  TOPCY,7X,
   3 9HEPH TOPCY,8X,11HDELTA TOPCY///)
901  FORMAT (I7,5F17.7,F17.2,F17.7/7X,7F17.7///)
902  FORMAT(////30X,16HMAXIMUM NUMBER (,I3,55H) OF ITERATIONS PERFORME
   1D.  CONVERGENCE NOT GUARANTEED.)
904  FORMAT(//,40X,36HINACCESSIBLE NODE ON MOP CALLED FOR.)
905  FORMAT(40X,55HNODE PLACED AT BEGINNING OF INACCESSIBLE REGION ON
   1MOP.///)
907  FORMAT(40X,49HNODE PLACED AT END OF INACCESSIBLE REGION ON MOP.//
   1/)
908  FORMAT(40X,55HNECESSARY NODE ON MOP IS INACCESSIBLE.  NO CONVERGE
   1NCE.//)
909  FORMAT(//20X,91HCHANGE IN INSERTION TIME EXCEEDS 12 HOURS.  SIGN
   1WILL BE CONSTRAINED TO THAT OF NEXT VALUE.//)
910  FORMAT(//25X,79HORBIT COAST ARC EXCURSION.  ARC WILL BE CONSTRAIN
   1ED TO WITHIN PI OF NEXT VALUE.///)
911  FORMAT(//20H TLI VARIABLES .....,
   1 10X,28HDELTA  = 0  (BY DEFINITION) ,15X,12HMHAT DECL  =,F14.8/
   2 30X,8HSIGMA  =,F13.8,22X,12HMHAT RYTAS =,F14.8/
   3 30X,8HALFA   =,F13.8,22X,12HW =  C3/2  =,F14.2,11H  (INFPS)SQ/
   4 30X,8HETA    =,F13.8,30X,4HC3 =,F14.2,11H  (INFPS)SQ/
   5 30X,8HAFNTMH =,F13.8,30X,4HC3 =,F14.9,11H   (KMPS)SQ/
   6 30X,8HCHAR V =,F13.4,7H  INFPS,23X,4HC3 =,F14.9,11H   (ERPH)SQ/
   7 30X,8HTTW    =,F13.8)
912  FORMAT(////44X,13HTLI BEGINNING,7X,7HPERIGEE,9X,10HTLI CUTOFF//
   1 30X,9HTIME      ,3F17.7/
   2 30X,9HDECL      ,3F17.7/
   3 30X,9HRYTAS     ,3F17.7/
   4 30X,9HAZM       ,3F17.7/
   5 30X,9HLONG      ,3F17.7/
   6 30X,9HR (N.M.)  ,3F17.7/
   7 30X,9HV (INFPS) ,3F17.7/
   8 30X,9HGAMMA     ,34X,F17.7///)
913  FORMAT(1H1/5X,7HYEAR  =,I10,11X,6HFL01 =,F13.7,8X,2HR1,5X,1H=,
   1 F13.5,3X,21HXPC (MOP LONGITUDE) =,F13.7/
   2 6X,7HDAY   =,I10,11X,6HC1    =,F13.7,8X,8HORBNUM =,I6,15X,
   3 21HYPC  (MOP LATITUDE) =,F13.7/
   4 6X,7HHOUR =,F13.7,8X,6HAZ1   =,F13.7,8X,8HFIV    =,F13.7,8X,
   5 3HRPC,17X,1H=,F13.5/
   6 6X,7HRAGBT =,F13.7,8X,6HA1    =,F13.7,8X,8HPERIOD =,F13.7)
914  FORMAT (6X,7HRNVBT =,F13.7,8X,6HB1    =,F13.7,8X,8HIPERT  =,I6,
   1 15X,6HWINDOW,14X,12H=   ATLANTIC)
915  FORMAT (6X,7HRNVBT =,F13.7,8X,6HB1    =,F13.7,8X,8HIPERT  =,I6,
   1 15X,6HWINDOW,14X,12H=    PACIFIC)
     END
```

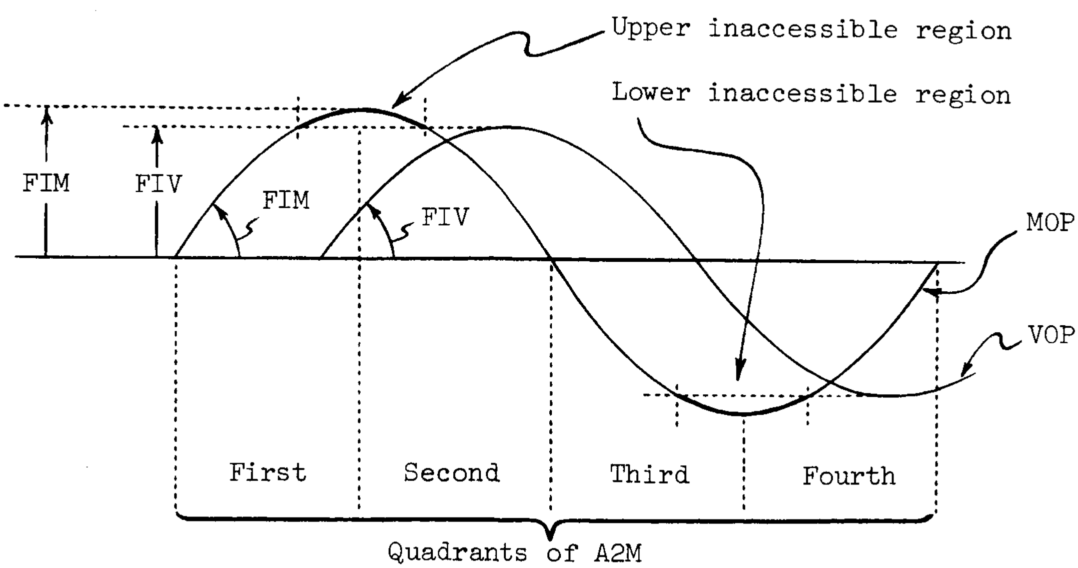
1.5 Specialized Logic Blocks In CIST for Special Problems

1.5.1 Inaccessible node logic.- The position of a "required" node between the MOP and the VOP is defined in the MOP by the angle A2M. This position of the node is "required" in order to satisfy requirements of orbit coast and trajectory flight times. The angle A2M is determined by the angle BETA and a position of pericynthion nadir defined by the angle A6. The value of A6 is a function of the presently defined time of pericynthion (TOPCY).

The declination of the required node is C2. If the inclination (FIV) of the VOP to the earth's equatorial plane is less than the absolute value of C2, a node between the MOP and the VOP cannot be achieved at the required position. The detection of the condition FIV < $|C2|$ causes control to go to the inaccessible node logic block.

The condition FIV < FIM is necessary but not sufficient for the condition FIV < $|C2|$ to arise. The condition FIV < FIM will occur during certain periods in certain years when the parking orbit is initiated with a 90° launch azimuth from Cape Kennedy.

The following figure shows that, when FIV < FIM, there are two regions in the MOP which cannot be "reached" by the VOP. These two regions will be referred to as the upper and lower inaccessible regions.



The upper inaccessible region is equally divided between the first and second quadrants of A2M. The lower inaccessible region is equally divided between the third and fourth quadrants of A2M.

The inaccessible node logic is controlled by the index IEX, which is actually a count of the number of times this logic is called. Prior to the beginning of the first iteration in the general logic, IEX is defined as 0. Each time this logic is called, IEX is stepped by 1.

Each time this logic is called, the externally defined inaccessible node is redefined at either the beginning or the end of the inaccessible region, such that it can be reached by the VOP. The declination of this redefined node will always have the magnitude of FIV and the sign of the originally inaccessible declination. The redefinition of C2 occurs at the beginning of the logic block. The value of A2M of the externally-defined, inaccessible node is stored as BADA2M for the correction of time of pericynthion at the end of the logic.

The quadrant of A2M wherein the redefined node lies is controlled by the index IOQ.

If IOQ = 1, the redefined node will be in the first or fourth quadrants of A2M. (The redefined node will be at either the beginning of the upper inaccessible region or at the end of the lower inaccessible region.)

If IOQ = 2, the redefined node will be in the second or third quadrants of A2M. (The redefined node will be at either the end of the upper inaccessible region or at the beginning of the lower inaccessible region.)

Each time this logic is called, IOQ is initially defined as 1, and subsequent tests within the logic determine whether it should be set equal to 2.

The first time this logic is called (IEX = 1), the node is redefined at the beginning of the inaccessible region (GO TO 620) and a statement to this effect is printed out. If within the subsequent iterations in the general logic, the node does not "jump over" the inaccessible region or "back away" from it, and a required position of the node again falls in the inaccessible region, this logic will be called a second time.

The second time this logic is called (IEX = 2), the node is redefined at the end of the inaccessible region (GO TO 630), and a statement to this effect is printed out. If within the subsequent iterations in the general logic, the node does not "stay ahead" of the inaccessible region and a required position of the node again falls in the inaccessible region, the logic will be called a third time.

The third time this logic is called (IEX = 3), the situation is considered hopeless, a statement to this effect is printed out, the convergence index (COI) is set equal to 3.0, and control is returned to the program calling CIST.

After C2 and IOQ are defined when the logic is called either the first or second times, the angles A2M, B2M, and AZ2M of the node are defined (statement 640) by calling GEOLAT. By calling GEOARG, RA2 is calculated (and AZ2M and B2M are redundantly calculated).

The presently defined time of pericynthion and associated variables in the XMS array will be compatible with the original inaccessible node, but will be incompatible with the redefined accessible node. Since the XMS variables are used in the empirical equations defining the simulated trajectory, these XMS variables should be "corrected" before returning to the general logic. In most cases, these corrections are expected to be insignificant; however, marginal cases can arise where the absence of these corrections can result in nonconvergence (COI = 3).

The correction in the time of pericynthion is calculated as the difference between the A2M of the redefined node and that of the original inaccessible node, stored as BADA2M, divided by the angular velocity (WM) of the moon in its orbit. Two assumptions are made: (1) that WM is a true average over the interval of time correction, and (2) that BETA will be constant such that a change in the angular position of the node in the MOP will result in an equal change in the position of pericynthion nadir.

With a corrected value of TOPCY, PERCYN is called, thus defining new values in the XMS array. Before subroutines ENERGY and FLYTYM are called, a correct value of FIVTL is calculated which is compatible with the redefined accessible node.

Control then returns to the general logic at statement 650.

```
      IF(IEX.EQ.0) IEX=1
      C2=SIGN(FIV,C2)
      BADA2M=A2M
      IOQ=1
      WRITE(6,904)
      GO TO (620,630,610),IEX
 610  WRITE(6,908)
      COI=3.0
      RETURN
 620  IF(C2.LT.0.0) IOQ=2
      IEX=2
```

(listing continued on next page)

```
      WRITE(6,906)
      GO TO 640
630   IF(C2.GT.0.0) IOQ=2
      IEX=3
      WRITE(6,907)
640   CALL GEOLAT (FIM,C2,DUM1,IOQ,A2M,B2M,AZ2M,DUM2)
      CALL GEOARG (FIM,A2M,RNM,AZ2M,B2M,DUM1,RA2)
      TCOR=A2M-B4DA2M
      CALL HELP (TCOR)
      TOPCY=TOPCY+TCOR/WM
      CALL PERCYN (TOPCY)
      FIVTL=AZ2M-PI/2.0
      CALL FLYTYM (IPERT,RQDFT)
      CALL ENERGY (IPERT,C2)
```

1.5.2 Orbit coast time excursion logic.- This logic is called when the absolute value of the angle CARC, from the parking orbit pickup state vector to the beginning of the TLI maneuver, changes by more than π between consecutive iterations in the general logic. This situation arises in the aforementioned iterative process when the position of the beginning of the TLI maneuver moves across the position of the orbit pickup state vector in the VOP, resulting in a change in CARC of almost 2π.

The successful convergence of an iteration will not be prevented if this happens only once or even several times within the iteration. However, the possibility exists of a condition of self-sustaining oscillation to be set up where the position of the beginning of TLI moves back and forth across the position of the orbit pickup state vector and convergence with the normal general logic is impossible. The probability of this type of oscillatory condition arising is very small, but finite; it has been observed to happen.

The orbit coast time excursion logic is controlled by the index IAIO. Before the beginning of the first iteration in the general logic, IAIO is set equal to 1. The first time this logic is called, IAIO is set equal to 2 and thereafter its value is not altered until the iteration process is terminated.

The logic itself is quite simple. The first time it is called CCARC, a control value of CARC is defined as equal to the present value of CARC. Every subsequent time the logic is called, the immediate value of CARC is changed by either $+2\pi$ or -2π such that CARC will be within π of CCARC.

Control is returned to the general logic at statement 657.

```
      GO TO (655,656),IAIO
655   IAIO=2
      CCARC=CARC
      WRITE (6,910)
      GO TO 657
656   SDAIO=1.0
      IF(CARC.GT.CCARC) SDAIO=-1.0
      CARC=CARC+SDAIO*2.0*PI
```

1.5.3 Time of orbit state vector excursion logic.- This specialized logic is called when the time of the parking orbit pickup state vector (T1) changes by more than 12 hours from the value in the preceding iteration (stored as PT1).

T1 is measured relative to base time, an input PRE(3) G.m.t. midpoint of the 24-hour period within which T1 is to occur. T1 is calculated as the difference between RNVBT and RNV divided by WE, where RNV is the right ascension of the ascending node of the VOP as presently oriented, RNVBT is the right ascension of the ascending node of the VOP when insertion occurs at base time (T1 = 0.0), and WE is the angular velocity of the earth's rotation. This difference between RNVBT and RNV must be defined between $-\pi$ and π if T1 is to be constrained within 12 hours of base time; i.e., within the prescribed 24-hour period.

When RNV is close to 180° from RNVBT, a small change in RNV can result in a change of almost 24 hours in T1 between two consecutive iterations. The successful convergence of an iteration will not be prevented if this happens only once or even several times within an iteration. However, the possibility exists of a condition of self-sustaining oscillation to be set up, where RNV oscillates back and forth over a very small range centered at 180° from RNVBT, and convergence with the normal general logic is impossible. The probability of this type of oscillatory condition arising is very small, as is the probability of the oscillation of CARC. But this probability is finite, and the oscillation of RNV has been observed to prevent successful convergence in the absence of a specialized logic designed to deal with this problem.

The T1 excursion logic is controlled by the index IOS, which is set equal to 1 in the general logic before beginning the first iteration. The first time this logic is called, IOS is set equal to 2 and thereafter its value is not altered until the iteration process is terminated.

The first time the logic is called, a corrective term (TOLCOR) for T1 is defined having the magnitude of a sidereal day and the sign of the present value of T1. Thereafter, every time the logic is called, TOLCOR is added to the immediate value of T1, and as a result T1 is always constrained to have the same sign as the value of T1 when the logic was first called. A statement to this effect is printed out the first time this logic is called.

In effect, T1 is constrained within a time interval much shorter than 12 hours; but this time interval is not required to lie within the originally prescribed 24-hour period within which T1 is to occur. Consequently, when this logic is called, T1 can (and usually does) wander just outside of the prescribed 24-hour period. But this happens only because convergence cannot be otherwise achieved; i.e., there is no "solution" with T1 within the prescribed 24-hour period. When this logic is called and T1 does move outside of the prescribed 24-hour period to achieve convergence, the resultant T1 is most likely to occur after the 24-hour period. It is most unlikely that resultant T1 will ever occur before the prescribed 24-hour period due to the fact that the first iteration in the general logic is begun with the assumption of the earliest possible value of time of pericynthion (TOPCY), and thereafter RNV only moves forward.

Control returns to the general logic at statement 680.

```
      GO TO (560,570),IOS
560   IOS=2
      TOLCOR=SIGN(2.0*PI/WE,T1)
      WRITE(6,909)
      GO TO 680
570   T1=T1+TOLCOR
```

2.0 SUBROUTINES OF CIST

There are nine subroutines of the present version of CIST. These subroutines can be classified into the following four groups:

ENERGY FLYTYM SUBB SUBCL	These subroutines contain the empirical equations of the simulated trajectory. All of them have the SIMUL block in common. Input and output are both through this common block and their calling arguments.
TLIMP	This subroutine defines variables of the simulation of the TLI maneuver. Conic elements of the resulting trajectory are also defined. The SIMUL block is in common. Input and output are through this common block.
PERCYN	This subroutine defines the variables describing the positions of the sun, the pericynthion nadir vector, and the position and motion of the moon. The SIMUL block is in common. This subroutine obtains Cartesian coordinates of positions and velocities by calling subroutine JPLEPH. Input to PERCYN is through its calling argument, output is through common.
GEOARG GEOLAT HELP	These subroutines perform trigonometric calculations. They contain no empirical equations. Input and output are entirely through their calling arguments. They have nothing in common.

Listings of these subroutines and more detailed descriptions of them can be found on the following pages.

2.1 Subroutine PERCYN

2.1.1 Identification.-

PERCYN (Ephemeris Inquisitor)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.1.2 Purpose.- Subroutine PERCYN computes variables describing the positions of the sun, pericynthion nadir vector, and the position and motion of the moon.

2.1.3 Usage.-

2.1.3.1 Calling sequence: CALL PERCYN (TOPCY).

2.1.3.2 Arguments:

Parameter name	In/Out	Type	Description
TOPCY	In	Real	Time of pericynthion in hours relative to a base time defined and stored external to PERCYN

2.1.3.3 Label common: All of the variables in the SIMUL common block in subroutine PERCYN are the output of PERCYN. These ephemeris variables constitute the entire XMS array. All of these ephemeris variables described below are defined at the time TOPCY.

Location in SIMUL block	MNEMONIC	Description
31	EMR	Earth-moon radius, in nautical miles
32	EMRDOT	$\dot{EMR}$, first derivative of earth-moon radius with respect to time, in knots.
34	FIM	Inclination of moon's orbit plane (MOP) to earth's equatorial plane, in radians.
35	RNM	Right ascension of the ascending node of the MOP, in radians. $-\pi < RNM \leq \pi$
36	AM	Argument of the moon in the MOP past its ascending node on the earth's equator, in radians $-\pi < AM \leq \pi$
37	RAM	Right ascension of the moon, in radians $-\pi < RAM \leq \pi$
38	DECM	Declination of the moon, in radians

Location in SIMUL block	MNEMONIC	Description
39	A6	Argument of pericynthion nadir in MOP past moon's ascending node on earth's equator, in radians $-\pi < A6 \leq \pi$
40	RA6	Right ascension of pericynthion nadir, in radians $-\pi < RA6 \leq \pi$
41	C6	Declination of pericynthion nadir, in radians
42	B6	Longitude of pericynthion, measured in earth's equatorial plane from ascending node of MOP, in radians. $-\pi < B6 \leq \pi$
43	AZ6	Azimuth of MOP at pericynthion nadir, in radians.
44	WM	Angular velocity of the moon, in rad/hr.
45 46 47	XHM YHM ZHM	Cartesian components of the angular momentum vector of the moon, in e.r.^{2}/hr, relative to the earth's center.
51	SMOPL	Longitude of the sun in an earth-centered MOP coordinate system, measured counter-clockwise from the position of the moon, in radians. $-\pi < SMOPL \leq \pi$
52	SMOPD	Declination of the sun in an earth-centered MOP coordinate system, in radians.
53	AS	Argument of the sun in the ecliptic past its ascending node on the earth's equatorial plane, in radians. $-\pi < AS \leq \pi$
54	RAS	Right ascension of the sun, in radians $-\pi < RAS \leq \pi$

Location in SIMUL block	MNEMONIC	Description
55	DS	Declination of the sun, in radians
56	COI	Convergence index, indicates different types of trouble to program calling CIST. In PERCYN, if ephemeris trouble is encountered, COI is set equal to 2.0.

2.1.3.4 Sample usage: (Refer to "Calling Sequence" and "Label Common" above)

2.1.3.5 Storage required: Coding occupies 1037_8 (543_{10}) locations.

2.1.3.6 Error codes and diagnotics: If subroutine JPLEPH returns an error to PERCYN, the following message is printed: "THE XMS ARRAY WE DID NOT FILL, FOR ALL IS NOT WELL IN EPHEMERISVILLE"

2.1.4 Method.-

2.1.4.1 Statement of algorithms: The ephemeris subroutine JPLEPH is called at time TOPCY. In JPLEPH, vectors are defined which describe the position and velocity of the moon and the position of the sun relative to the earth's center. The sun's position vector is stored in the six-dimensioned array RS. The position and velocity vectors of the moon are stored in the nine-dimensioned array RM: the position vector, in locations 1, 2, and 3; the velocity vector, in locations 7, 8, and 9. In the following algorithm description, the moon's position vector will be denoted as $\overrightarrow{RM}$ and the moon's velocity vector will be denoted as $\overrightarrow{VM}$. The units of these vectors in the RS and RM arrays are in earth radii and hours.

If trouble of any kind is encountered in JPLEPH, a warning message is printed by PERCYN, COI is set equal to 2.0, and a return is executed to CIST.

If no such difficulty is encountered, EMR and EMRDOT are first calculated.

$$EMR = 3443.93358\ |\overrightarrow{RM}|$$

$$EMRDOT = 3443.93358 \left[\frac{\overrightarrow{RM} \cdot \overrightarrow{VM}}{|\overrightarrow{RM}|} \right]$$

RAM is calculated using a four quadrant arctangent function. RA6 is calculated by merely adding π to RAM and calling HELP to define it between π and $-\pi$.

$$RAM = \tan^{-1}(RM_y/RM_x)$$

$$RA6 = RAM + \pi$$

CALL HELP (RA6)

DECM is calculated using a two quadrant arctangent function. C6 is then defined as the negative of DECM.

$$DECM = \tan^{-1}\left[\frac{RM_z}{\sqrt{RM_x^{\,2} + RM_y^{\,2}}}\right]$$

$$C6 = -DECM$$

The cartesian components (XHM, YHM, ZHM) of the moon's angular momentum vector ($\vec{HM}$) are calculated by the straightforward cross product $\vec{HM} = \vec{RM} \times \vec{VM}$. The moon's angular velocity (WM) is then calculated as

$$WM = \frac{|\vec{HM}|}{|\vec{RM}|^2}$$

FIM is calculated using a two quadrant arctangent function

$$FIM = \tan^{-1}\left[\frac{\sqrt{XHM^2 + YHM^2}}{ZHM}\right]$$

RNM is calculated using a four quadrant arctangent function. B6 is then calculated as the difference (RA6 - RNM), defined between π and $-\pi$ by subroutine HELP.

$$RNM = \tan^{-1}(XHM/-YHM)$$

$$B6 = RA6 - RNM$$

CALL HELP (B6)

Knowing B6 and FIM, A6 is calculated using a four quadrant arctangent function, the argument of which is designed to give answers in the proper quadrant. AM is then calculated as A6 + π, defined between π and $-\pi$ by subroutine HELP.

$$A6 = \tan^{-1}\left[\frac{\tan B6}{\cos FIM}\right]$$

$$AM = A6 + \pi$$

CALL HELP (AM)

Knowing B6 and C6, AZ6 is calculated using a four quadrant arctangent function, the argument of which is designed to give answers in the proper quadrant.

$$AZ6 = \tan^{-1}\left[\frac{\tan |B6|}{|\sin C6|}\right]$$

The solar ephemeris variables RAS, DS, and AS, are calculated using four and two quadrant arctangent functions.

$$RAS = \tan^{-1}\left[RS_y/RS_x\right]$$

$$DS = \tan^{-1}\left[\frac{RS_z}{\sqrt{RS_x^{\ 2} + RS_y^{\ 2}}}\right]$$

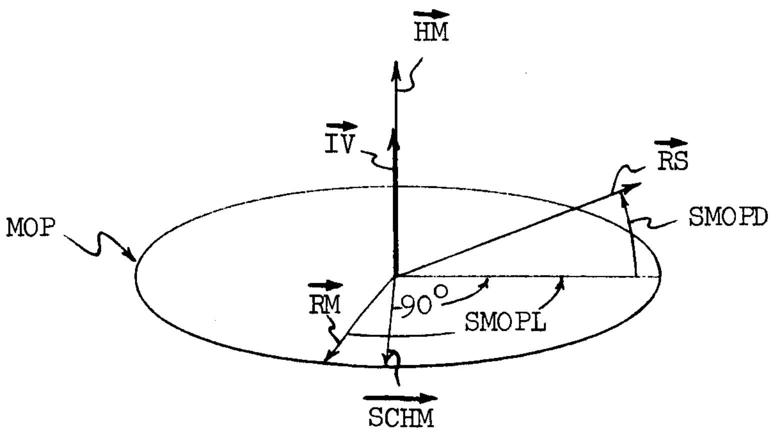
$$AS = \tan^{-1}\left[\sqrt{\frac{|\vec{RS}|^2}{RS_x^2} - 1}\right]$$

Quadrant allocation of AS is achieved by additional statements testing RS_x and RS_z.

The solar variables SMOPL and SMOPD are then calculated. In doing this, two intermediate vectors $\overrightarrow{SCHM}$ and $\vec{IV}$, are used.

$$\overrightarrow{SCHM} = \vec{RS} \times \vec{HM}$$

$$\vec{IV} = \vec{RM} \times \overrightarrow{SCHM}$$



In the above illustration $\vec{IV}$ is coincident with $\vec{HM}$.

First, using a four quadrant arctangent subroutine, SMOPL is calculated as the angle from $\vec{RM}$ to $\vec{SCHM}$.

$$\mathrm{SMOPL} = \tan^{-1}\left[\frac{|\vec{IV}|}{\vec{RM} \cdot \vec{SCHM}}\right]$$

This value of SMOPL will always be in either the first or second quadrants. If the sign of ($\vec{IV} \cdot \vec{HM}$) is negative, this angle should be in the third or fourth quadrant. In this case, the sign of this value of SMOPL is made negative. With the angle from $\vec{RM}$ to $\vec{SCHM}$ defined in the proper quadrant, SMOPL is then calculated as the aforementioned angle (stored in SMOPL) plus $\pi/2$ and then defined between π and $-\pi$ by subroutine HELP.

$$\mathrm{SMOPL} = \mathrm{SMOPL} + \pi/2$$

CALL HELP (SMOPL)

SMOPD is then calculated

$$SMOPD = \frac{\pi}{2} - \tan^{-1}\left[\frac{|\overrightarrow{SCHM}|}{\overrightarrow{RS} \cdot \overrightarrow{HM}}\right]$$

Return is then executed to the program calling PERCYN.

2.1.4.2 Derivations or references: See reference 4.

2.1.5 Restrictions.-

2.1.5.1 Range of numbers that can be processed: Input is restricted in that only the years 1950 through 1999 can be interrogated on the ephemeris tape. Other numerical limits will be imposed by the system FORTRAN library functions.

2.1.5.2 Range of applicability: Providing that the ephemeris has been properly initialized by an external driver, PERCYN can be used by almost any other program assuming the user has the storage array definitions. In a general sense, the argument TOPCY can be any time relative to a pre-established base time.

2.1.5.3 Other programs required: Subroutines JPLEPH and HELP.

2.1.6 Accuracy.- See "Restrictions" above.

2.1.7 Coding Information.- All calculations are single precision. The only exception is that JPLEPH requires double precision arguments.

2.1.8 Listing.-

```
SUBROUTINE PERCYN (TOPCY)
COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
DOUBLE PRECISION T,RS(6),RM(9),PNL(3,3)
EQUIVALENCE (XMS(1),EMR)
EQUIVALENCE (XMS(2),EMRDOT)
EQUIVALENCE (XMS(4),FIM)
EQUIVALENCE (XMS(5),RNM)
EQUIVALENCE (XMS(6),AM)
EQUIVALENCE (XMS(7),RAM)
EQUIVALENCE (XMS(8),DECM)
EQUIVALENCE (XMS(9),AS)
```

(Listing continued on next page)

```
      EQUIVALENCE (XMS(10),RA6)
      EQUIVALENCE (XMS(11),C6)
      EQUIVALENCE (XMS(12),B6)
      EQUIVALENCE (XMS(13),AZ6)
      EQUIVALENCE (XMS(14),WM)
      EQUIVALENCE (XMS(15),XHM)
      EQUIVALENCE (XMS(16),YHM)
      EQUIVALENCE (XMS(17),ZHM)
      EQUIVALENCE (XMS(21),SMOPL)
      EQUIVALENCE (XMS(22),SMOPD)
      EQUIVALENCE (XMS(23),AS)
      EQUIVALENCE (XMS(24),RAS)
      EQUIVALENCE (XMS(25),DS)
      EQUIVALENCE (TAR(1),COI)
      C=3443.93358
      PI=3.1415927
      T=TOPCY
      CALL JPLEPH (0,T,1,RS,RM,PNL,IERR)
      IF(IERR.EQ.0) GO TO 20
      WRITE (6,900)
      COI=2.0
      RETURN
20    EMR=RM(4)*C
      EMRDOT=(RM(1)*RM(7)+RM(2)*RM(8)+RM(3)*RM(9))*C/RM(4)
      RAM=ATAN2(RM(2),RM(1))
      RA6=RAM+PI
      CALL HELP (RA6)
      DECM=ATAN2(RM(3),SQRT(RM(1)*RM(1)+RM(2)*RM(2)))
      C6=-DECM
      XHM=RM(2)*RM(9)-RM(3)*RM(8)
      YHM=RM(3)*RM(7)-RM(1)*RM(9)
      ZHM=RM(1)*RM(8)-RM(2)*RM(7)
      WM=SQRT(XHM*XHM+YHM*YHM+ZHM*ZHM)/RM(5)
      FIM=ATAN(SQRT(XHM*XHM+YHM*YHM)/ZHM)
      RNM=ATAN2(XHM,(-YHM))
      B6=RA6-RNM
      CALL HELP (B6)
      A6=ATAN2(SIN(B6),COS(B6)*COS(FIM))
      AM=A6+PI
      CALL HELP (AM)
      AZ6=ATAN2(ABS(SIN(B6)),COS(B6)*ABS(SIN(C6)))
      RAS=ATAN2(RS(2),RS(1))
      DS=ATAN2(RS(3),SQRT(RS(1)*RS(1)+RS(2)*RS(2)))
      AS=ATAN(SQRT(RS(5)/(RS(1)*RS(1))-1.0))
      IF(RS(1).LT.0.0) AS=PI-AS
      IF(RS(3).LT.0.0) AS=-AS
```

(Listing continued on next page)

```
      XSCHM=RS(2)*ZHM-RS(3)*YHM
      YSCHM=RS(3)*XHM-RS(1)*ZHM
      ZSCHM=RS(1)*YHM-RS(2)*XHM
      XIV=RM(2)*ZSCHM-RM(3)*YSCHM
      YIV=RM(3)*XSCHM-RM(1)*ZSCHM
      ZIV=RM(1)*YSCHM-RM(2)*XSCHM
      SMOPL=ATAN2(SQRT(XIV*XIV+YIV*YIV+ZIV*ZIV),
     1 RM(1)*XSCHM+RM(2)*YSCHM+RM(3)*ZSCHM)
      IF (XIV*XHM+YIV*YHM+ZIV*ZHM.LT.0.0) SMOPL=-SMOPL
      SMOPL=SMOPL+PI/2.0
      CALL HELP (SMOPL)
      SMOPD=PI/2.0-ATAN2(SQRT(XSCHM*XSCHM+YSCHM*YSCHM+ZSCHM*
     1 ZSCHM),RS(1)*XHM+RS(2)*YHM+RS(3)*ZHM)
      RETURN
  900 FORMAT(///30X,70HTHE XMS ARRAY WE DID NOT FILL,
     1 FOR ALL IS NOT WELL IN EPHEMERISVILLE.)
      END
```

2.2 Subroutine ENERGY

2.2.1 Identification.-

ENERGY (Trajectory Simulation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.2.2 Purpose.- Subroutine ENERGY computes the energy at perigee of a specific simulated trajectory.

2.2.3 Usage.-

2.2.3.1 Calling sequence: Call ENERGY (IPERT, C4)

2.2.3.2 Arguments:

Parameter name	In/Out	Dimension	Type	Description
IPERT	In	-	Integer	Perturbation instruction index =0, no perturbations =1, only earth oblateness considered =2, only solar gravitation considered =3, both earth oblateness and solar gravitation are considered
C4	In	-	Real	Declination of perigee (radians)

2.2.3.3 Label common:

Location in SIMUL block	MNEMONIC	Description
13	XPC	Longitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, measured counterclockwise from the extension of the earth-moon axis on the back side of the moon, in degrees.
14	YPC	Latitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, in degrees.
15	RPC	Radius of pericynthion of the simulated trajectory relative to the moon's center, in nautical miles
31	EMR	Earth-moon radius at the time of pericynthion of the simulated trajectory, in nautical miles

Location in SIMUL block	MNEMONIC	Description
32	EMRDOT	$\dot{EMR}$, the first derivative of earth-moon radius with respect to time, in knots.
51	SMOPL	Longitude of the sun in an earth-centered MOP coordinate system, measured counterclockwise from the position of the moon at the time of pericynthion of the simulated trajectory, in radians.
61	FIVTL	Inclination of the trajectory plane to the MOP, defined at perigee. FIVTL is defined as negative if the trajectory is going below the MOP, in radians.
66	W	Energy at perigee of the simulated trajectory, in (international ft/sec)^2. $W = \frac{V^2}{2} - \frac{\mu}{r}$
88	R4	Radius of perigee of the simulated trajectory relative to the earth's center, in nautical miles

2.2.3.4 Sample usage: Refer to "Calling Sequence" and "Label Common" above.

2.2.3.5 Storage required: Coding occupies $577_8 (383_{10})$ locations.

2.2.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

2.2.4 Method.-

2.2.4.1 Statement of algorithm: The effects of solar gravitation (SEW) and earth oblateness (EOBW) on trajectory energy (W) at perigee, are first calculated.

$$SEW = [3.9 \sin(2\ SMOPL - 1.657) - 1.3] \times 10^4$$

$$EOBW = (5.97 - 2.67 \cos C4)\ (10^5 \cos\ 2.55\ C4)$$

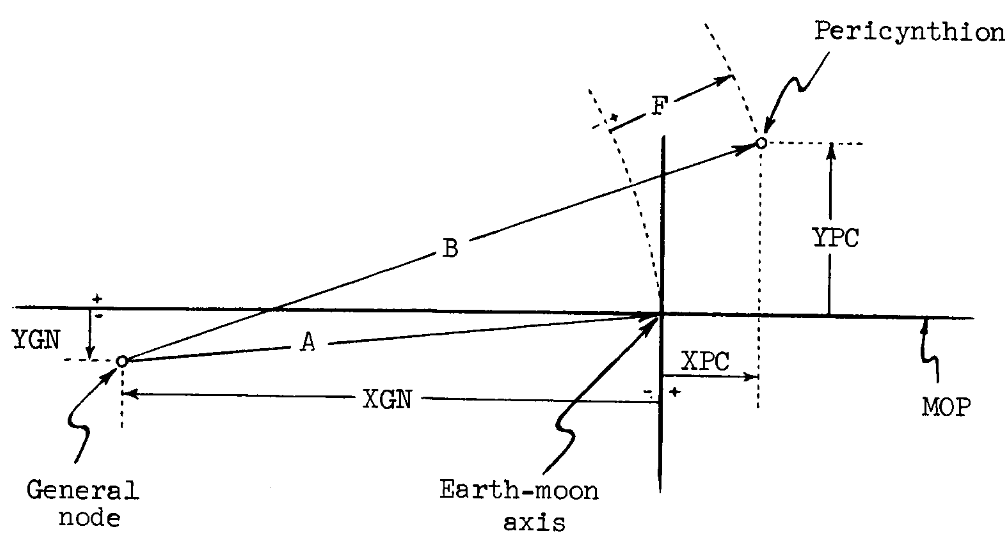
After each of these calculations, IPERT is tested to see whether the effect of the perturbation is to be considered in the final value of W. If it is not to be considered, the value of the perturbation effect (SEW or EOBW) is set equal to zero.

Two intermediate variables, PHI and FI, are defined for use in the final equation for W. These two variables have no known physical significance.

$$PHI = \tan\left(\frac{30 + \frac{2}{3}\,XPC}{57.29578}\right)$$

$$FI = \frac{1}{EMR - 29000.0}$$

The angle F is next computed for use in the final equation for W. As described in reference 1, F is the difference between two angles, A and B, measured from the general node. As shown in the following figure, A is measured to the earth-moon axis, and B is measured to pericynthion. The longitude, XGN, and latitude, YGN, of the general node are given by empirical equations which are approximations.



$$XGN\ (deg) = -68.0 + \frac{EMR}{10000.0} + 0.625\ XPC$$

$$YGN\ (rad) = -\sin FIVTL \left[\frac{9.6 - 0.16\ (XGN + 48.0)}{57.29578} \right]$$

$$XGN\ (rad) = XGN\ (deg)\ /57.29578$$

The calculation of A and B are straightforward problems in spherical trigonometry. Two additional intermediate variables involved in these calculations are as follows:

$$COA = \cos A$$

and

$$COB = \cos B.$$

F is then given, in degrees, by the simple equation

$$F = 57.29578\ (B-A)$$

The value of W is then calculated using the following equation:

$$\begin{aligned} W = {} & -2714728.4 - (1.4002127 \times 10^{12})FI + EMRDOT\ [3070.6244 \\ & + (2.1470226 \times 10^{8})FI + 1.1495569\ EMRDOT] + 750.\ R4 \\ & + (1.0 - \cos FIVTL)(2.2\ EMR - 6.8 \times 10^{4}) + F(-75150. \\ & - .05\ EMR - 8.75\ EMRDOT) - [4050. \\ & + 364500./(F-90.)]\ (2.25\ EMRDOT - .01\ EMR + 7070.) \\ & + (8.6016 \times 10^{6})\ PHI^{2} \Big[1./[(RPC - 1015.)/(2217.025\ PHI) \\ & + 1.] -1. \Big] + SEW + EOBW \end{aligned}$$

2.2.4.2 Derivation or references: See reference 1.

2.2.5 Restrictions.-

2.2.5.1 Range of numbers that can be processed: The only requirement of the input ephemeris variables is that they represent realistic earth-moon-sun conditions. The input trajectory variables must represent a translunar trajectory of the type described in the reference.

2.2.5.2 Range of applicability: Subroutine ENERGY is designed for use with the CIST package. However, ENERGY can be used in other applications of simulated trajectories.

2.2.5.3 Other programs required: No other programs are required.

2.2.6 Accuracy.- The greatest accuracy of the perigee energy equation in this subroutine, will be obtained if RPC is within several hundred nautical miles of 1015 n. mi., and R4 is within several tens of nautical miles of 3550 n. mi. The constraint of RPC is at present the more serious limitation. Usable values of W are given by this subroutine for RPC's of several thousand nautical miles with reasonable consistency. Greater accuracy will also be achieved the closer XPC and YPC are to zero; however, these restrictions of XPC and YPC are not nearly as critical as the limitations of RPC and R4. Reasonable accuracy should be expected if XPC is kept between 20° and -50° and YPC is kept between 10° and -10°. Greater accuracy should be expected the closer |FIVTL| is to zero.

2.2.7 Coding information.- All calculations and input/output are in single precision.

2.2.8 Listing.-

```
      SUBROUTINE ENERGY (IPERT,C4)
      COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
      EQUIVALENCE (PRE(13),XPC)
      EQUIVALENCE (PRE(14),YPC)
      EQUIVALENCE (PRE(15),RPC)
      EQUIVALENCE (XMS(1),EMR)
      EQUIVALENCE (XMS(2),EMRDOT)
      EQUIVALENCE (XMS(21),SMOPL)
      EQUIVALENCE (TAR(6),FIVTL)
      EQUIVALENCE (TAR(11),W)
      EQUIVALENCE (TRAJ(8),R4)
C
C     SEW  = EFFECT OF SOLAR GRAVITATION
C
      SEW=3.9E+04*SIN(2.0*SMOPL-1.657)-1.3E+04
```

(Listing continued on next page)

```
IF(IPERT.EQ.0.OR.IPERT.EQ.1)  SEW=0.0

EOBW = EFFECT OF EARTH OBLATENESS

EOBW=(5.97-2.67*COS(C4))*COS(2.55*C4)*1.0E+05
IF(IPERT.EQ.0.OR.IPERT.EQ.2)  EOBW=0.0


DPR=57.295780
PHI=(30.0+XPC*0.66666667)/DPR
PHI=SIN(PHI)/COS(PHI)
FI=1.0/(EMR-29000.0)
XGN=-58.0+EMR/10000.0+0.625*XPC
YGN=-SIN(FIVTL)*(9.5-0.15*(XGN+48.0))/DPR
XGN=XGN/DPR
COA=COS(XGN)*COS(YGN)
A=ATAN(SQRT(1.0/(COA*COA)-1.0))
COB=SIN(YGN)*SIN(YPC/DPR)+COS(YGN)*COS(YPC/DPR)*COS(XPC/DPR-XGN)
B=ATAN(SQRT(1.0/(COB*COB)-1.0))
F=(B-A)*DPR
W=-2714728.4-1.4002127E+12*FI+EMRDOT*(3070.6244+
1 FI*2.1470225E+06+EMRDOT*1.1495569)
2 +R4*750.0+(1.0-COS(FIVTL))*(2.2*EMR-6.8E+04)
3 +F*(-75150.0-0.05*EMR-EMRDOT*8.75)
4 -(4050.0+354500.0/(F-90.0))*(EMRDOT*2.25-EMR*0.01+7070.0)
5 +8.501500E+06*PHI**2*(1.0/((RPC-1015.0)/(2217.0250*PHI)+1.0)-1.0)
6 +SEW+EOBW
RETURN
END
```

2.3 Subroutine FLYTYM

2.3.1 Identification.-

FLYTYM (Trajectory Simulation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.3.2 Purpose.- Subroutine FLYTYM calculates the perigee to pericynthion flight time of a specific simulated translunar trajectory.

2.3.3 Usage.-

2.3.3.1 Calling sequence: CALL FLYTYM (IPERT, RQDFT)

2.3.3.2 Arguments:

Parameter name	In/out	Dimension	Type	Description
IPERT	In	-	Integer	Perturbation instruction index =0, no perturbations considered =1, only earth oblateness considered =2, only solar gravitation considered =3, both earth oblateness and solar gravitation considered
RQDFT	Out	-	Real	Flight time from perigee to pericynthion on the simulated translunar trajectory in hours

2.3.3.3 Label common:

Location in SIMUL block	MNEMONIC	Description
13	XPC	Longitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, measured counterclockwise from the extension of the earth-moon axis on the back side of the moon, in degrees.
14	YPC	Latitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, in degrees.
15	RPC	Radius of pericynthion of the simulated trajectory relative to the moon's center, in nautical miles.

Location in SIMUL block	MNEMONIC	Description
31	EMR	Earth-moon radius at the time of pericynthion of the simulated trajectory, in nautical miles.
32	EMRD	$\dot{EMR}$, the first derivative of earth-moon radius with respect to time, in knots.
51	SMOPL	Longitude of the sun in an earth-centered MOP coordinate system, measured counterclockwise from the position of the moon at the time of pericynthion of the simulated trajectory, in radians.
61	FIVTL	Inclination of the trajectory plane to the MOP, defined at perigee. FIVTL is defined as negative if the trajectory is going below the MOP, in radians.

2.3.3.4 Sample usage: Refer to "Calling Sequence" and "Label Common" above.

2.3.3.5 Storage required: Coding occupies 457_8 (303_{10}) locations.

2.3.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

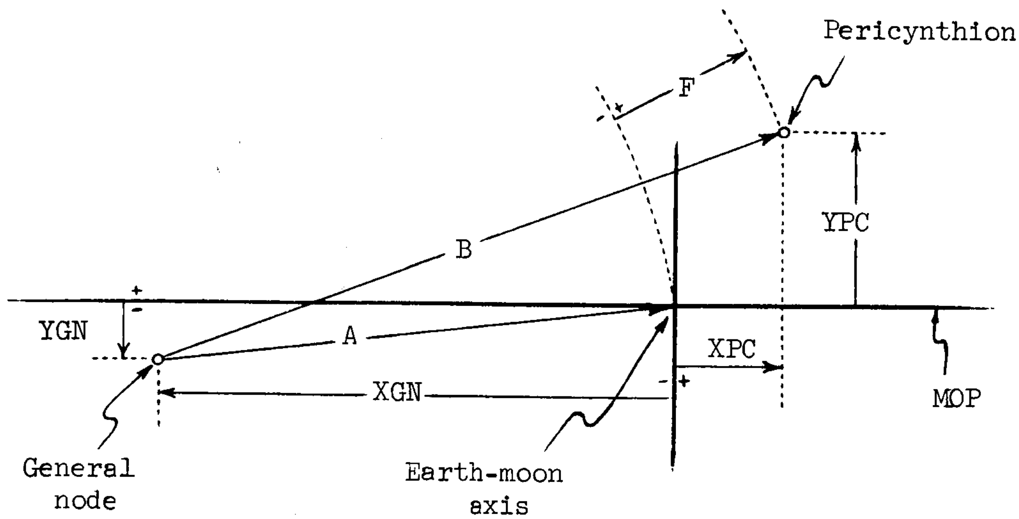
2.3.4 Method.-

2.3.4.1 Statement of algorithms: The effect of solar gravitation (SGFT) on flight time is first calculated.

$$\text{SGFT} = (\text{EMR})(2 \times 10^{-7}) + \cos\,(0.1745 - 2\ \text{SMOPL})\Big[(\text{EMR})(9.6 \times 10^{-7}) - (\text{EMRD})(7.5 \times 10^{-5}) - 0.1554\Big] - 0.028$$

IPERT is then tested to see if the effect of this perturbation is to be considered in the final value of flight time (RQDFT). If it is not to be considered (IPERT = 0 or 1), SGFT is set equal to zero.

The angle F is next calculated for use in the flight time equation. As described in the reference, F is the difference between two angles, A and B, measured from the general node. As shown in the following figure, A is measured to the earth-moon axis, and B is measured to pericynthion. The longitude, XGN, and latitude, YGN, of the general node are given by approximate empirical equations.



$$XGN(deg) = -68.0 + \frac{EMR}{10\ 000.0} + 0.625\ XPC$$

$$YGN(rad) = (-\sin FIVTL)\ \frac{9.6 - 0.16\ (XGN + 48.0)}{57.29578}$$

$$XGN(rad) = XGN/57.29578$$

The calculation of A and B are straightforward problems in spherical trigonometry. Two intermediate variables involved in these calculations are as follows:

$$COA = \cos A$$

$$COB = \cos B$$

F is then given, in degrees, by the simple equation

$$F = 57.29578\ (B - A)$$

The perigee-to-pericynthion flight time of the simulated trajectory is then calculated using a lengthy empirical equation. This flight time is given the address RQDFT because in the present CIST logic, this is the required flight time, as opposed to available flight time.

$$RQDFT = \Big[-23.480035 + (4.455251 \times 10^{-4})\ EMR$$
$$- EMRD\ [(1.6723713 \times 10^{-7})\ EMR - (1.5007662 \times 10^{-2})]$$
$$+ (\cos FIVTL - 1.0)\Big((2.4 \times 10^{-5})\ EMR - 3.96\Big)$$
$$- \Big(0.012 - (3.3 \times 10^{-6})\ EMR + (5.0 \times 10^{-4})\ EMRD\Big)^2$$
$$\Big(425.53191 + 1.0/\big[(2.209 \times 10^{-5})F/\big(0.048$$
$$- (1.32 \times 10^{-5})\ EMR + 0.002\ EMRD\big) - 0.00235\big]\Big)$$
$$+ SGFT\Big]\ (RPC/1015.0)^{0.17}$$

Having calculated the flight time, a return is given by the subroutine.

2.3.4.2 Derivations or references: See reference 1.

2.3.5 Restrictions.-

2.3.5.1 Range of numbers that can be processed: The only requirement of the input ephemeris variables is that they represent realistic earth-moon-sun conditions. The input trajectory variables must represent a translunar trajectory of the type of described in the reference.

2.3.5.2 Range of applicability: Subroutine FLYTYM is designed for use with the CIST package. However, FLYTYM can be used in other applications of simulated trajectories.

2.3.5.3 Other programs required: No other programs are required.

2.3.6 Accuracy.- The greatest accuracy of the flight time equation in this subroutine will be obtained if RPC is within several hundred nautical miles of 1015 n. mi., and perigee radius is within several tens of nautical miles of 3550 n. mi. The constraint of RPC is at present the more serious limitation. Usable values of RQDFT are given by this subroutine for RPC's of several thousand nautical miles with reasonable consistency. Greater accuracy will also be achieved the closer XPC and YPC are to zero; however, these restrictions of XPC and YPC are not nearly as critical as the limitation of RPC. Reasonable accuracy should be expected if XPC is kept between 20° and -50°, and YPC is kept between 10° and -10°. Greater accuracy should also be expected the closer |FIVTL| is to zero.

2.3.7 Coding information.- All calculations and input/output are in single precision.

2.3.8 Listing.-

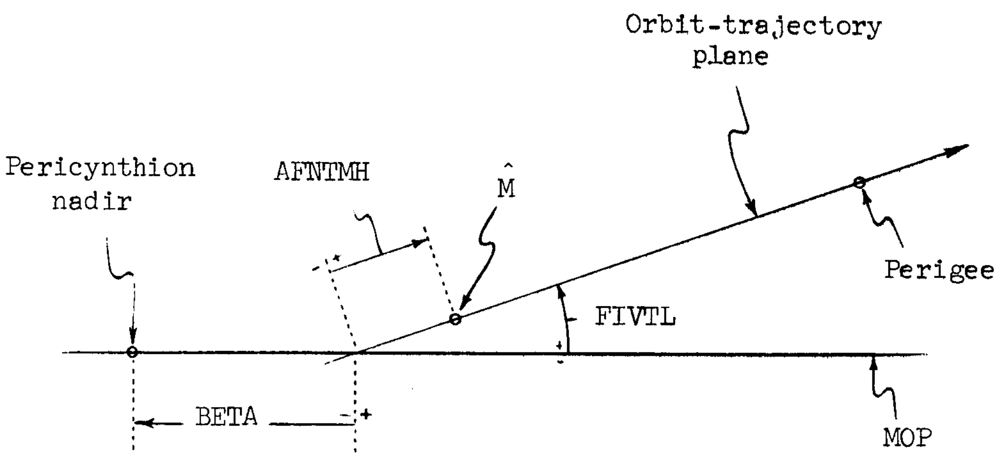
```
      SUBROUTINE FLYTYM (IPERT,RQDFT)
      COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
      EQUIVALENCE (PRE(13),XPC)
      EQUIVALENCE (PRE(14),YPC)
      EQUIVALENCE (PRE(15),RPC)
      EQUIVALENCE (XMS(1),EMR)
      EQUIVALENCE (XMS(2),EMRD)
      EQUIVALENCE (XMS(21),SMOPL)
      EQUIVALENCE (TAR(6),FIVTL)
      DPR=57.29578
C
C     SGFT = EFFECT OF SOLAR GRAVITATION
C
      SGFT=2.0E-07*EMR+COS(0.1745-2.0*SMOPL)*(EMR*9.6E-07-EMRD*7.5E-05
     1 -0.1554)-0.028
      IF(IPERT.LE.1) SGFT=0.0
      XGN=-68.0+EMR/10000.0+0.625*XPC
      YGN=-SIN(FIVTL)*(9.6-0.16*(XGN+48.0))/DPR
      XGN=XGN/DPR
      COA=COS(XGN)*COS(YGN)
      A=ATAN(SQRT(1.0/(COA*COA)-1.0))
      COB=SIN(YGN)*SIN(YPC/DPR)+COS(YGN)*COS(YPC/DPR)*COS(XPC/DPR-XGN)
      B=ATAN(SQRT(1.0/(COB*COB)-1.0))
      F=(B-A)*DPR
      RQDFT=(-23.480035+EMR*4.455251E-04-EMRD*(EMR*1.6723713E-07
     1 -1.5007662E-02)+(COS(FIVTL)-1.0)*(EMR*2.4E-05-3.96)
     2 -(0.012-3.3E-06*EMR+5.0E-04*EMRD)**2
     3 *(425.53191+1.0/(F*2.209E-05/(0.048-EMR*1.32E-05+EMRD*0.002)
     4 -0.00235))+SGFT)*(RPC/1015.0)**0.17
      RETURN
      END
```

2.4 Subroutine SUBB

2.4.1 Identification.-

SUBB (Trajectory Simulation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.4.2 Purpose.- Subroutine SUBB calculates two angles, BETA and AFNTMH, which are output in its calling argument. These two angles are shown in the following figure.



BETA is essential to the definition of the perigee state vector of the simulated trajectory in CIST. The angle AFNTMH (stands for angle from node to M-hat) is essential to the definition of TLI tarteting elements in CIST.

The calculations of BETA and AFNTMH are based upon the use of an empirical mechanism called the translunar injection tangency surface, which is described in detail in reference 3. The $\hat{M}$ TLI target vector is defined as the point of tangency of the orbit-trajectory plane on the tangency surface.

2.4.3 Usage.-

2.4.3.1 Calling sequence: CALL SUBB (IPERT, BETA, AFNTMH).

2.4.3.2 Arguments:

Parameter name	In/Out	Type	Description
IPERT	In	Integer	Perturbation instruction index =0, no perturbations considered =1, only earth oblateness considered =2, only solar gravitation considered =3, both earth oblateness and solar gravitation considered
BETA	Out	Real	The angle measured in the MOP from the node of the outgoing trajectory on the MOP to the pericynthion nadir. BETA is defined as negative when pericynthion nadir occurs behind the node, as is normally the case.
AFNTMH	Out	Real	Angle in the orbit-trajectory plane from the MOP node to the $\hat{M}$ TLI target vector

2.4.3.3 Label common:

Location in SIMUL block	MNEMONIC	Description
13	XPC	Longitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, measured counterclockwise from the extension of the earth-moon axis on the back side of the moon, in degrees.
14	YPC	Latitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, in degrees.
36	AM	Argument of the moon in the MOP, at the time of trajectory pericynthion, past the ascending node of the MOP on the earth's equator, in radians.

Location in SIMUL block	MNEMONIC	Description
55	DS	Declination of the sun at the time of trajectory pericynthion, in radians.
61	FIVTL	Inclination of the trajectory plane to the MOP, defined at perigee. FIVTL is defined as negative if the trajectory is going below the MOP, in radians.

2.4.3.4 Sample usage: Refer to "Calling Sequence" and "Label Common" above for usage.

2.4.3.5 Storage required: Coding occupies 471_8 (313_{10}) locations.

2.4.3.6 Error codes and diagnostics: If the latitude (YT) of the axis of mutual tangency is greater than |FIVTL|, the following message is written: "WARNING, TANGENCY SURFACE PROBLEM, YT DEFINED AS SIGN (ABS(FIVTL), YT)."

2.4.4 Method.-

2.4.4.1 Statement of algorithms: Detailed descriptions of the theory and use of the tangency surface are very lengthy and can be found in reference 3. The equations used in this subroutine are identical to those presented in the reference.

First, the longitude, XC, and latitude, YC, in the MOP coordinate system, of the center of the tangency surface lobe which will be used, are calculated in radians.

$$XC = \frac{1.93 - 0.037\ XPC}{57.29578}$$

$$YC = -\frac{\mathrm{SIGN}\ (1.0,\ FIVTL)(0.60 - 0.02\ XPC) - 0.035\ YPC}{57.29578}$$

The intermediate variable PLUS is next defined. PLUS is the equivalent of the product of Q and cos (AM - S) as described in reference 3. PLUS contains the effects of solar gravitation and earth oblateness. Consequently, the method used to define PLUS is dependent upon the perturbation instruction index, IPERT.

If IPERT = 0, PLUS = 0

If IPERT = 1, $\text{PLUS} = 0.0317 \cos \left[\text{AM} - \dfrac{18}{57.29578} \right]$

If IPERT = 2 or 3,

$$\text{PLUS} = \left(0.0285 + 0.0115 \cos (3.85\,\text{DS}) \right) \cos \left(\text{AM} - \frac{10 + 5 \cos (3.85\,\text{DS})}{57.29578} \right)$$

The latitude, YT, relative to the MOP of the node of mutual tangency of the two lobes of the tangency surface is next calculated in radians.

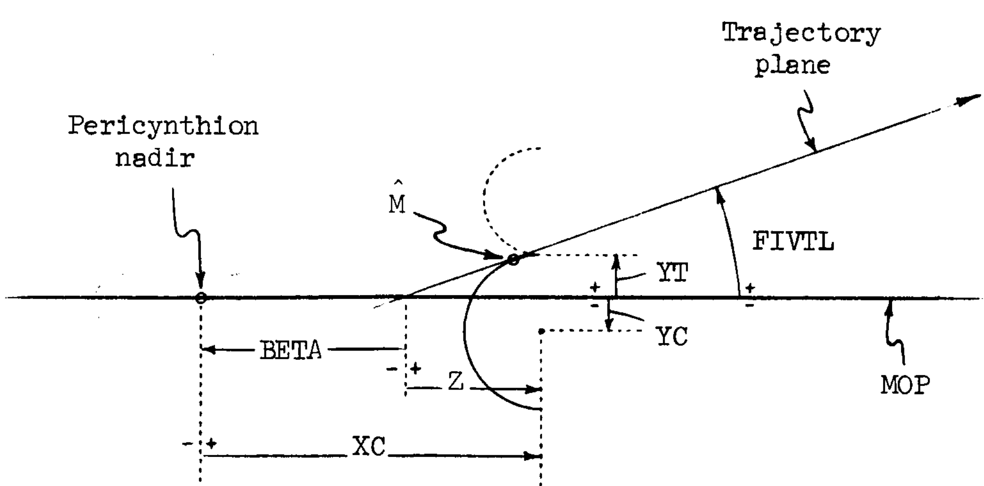
$$\text{YT} = \frac{\text{PLUS} - \text{YPC}\ (0.015165 - 0.000201\ \text{XPC})}{57.29578}$$

In order for tangency to be achieved, $|\text{FIVTL}| > |\text{YT}|$. A test is next made to see if this condition is satisfied. If this condition is not satisfied, YT is redefined as follows,

$$\text{YT} = \text{SIGN}\ (|\text{FIVTL}|,\ \text{YT})$$

such that tangency can be achieved.

Prerequisite to the calculation of BETA and AFNTMH is the calculation of the angle Z, shown in the following figure. The negative of the sine of Z is denoted as S.



S is normally found by straightforward spherical trigonometry.

$$S = \frac{-\sin(YT - YC) - \cos FIVTL \sin YC}{\sin FIVTL \cos YC}$$

and Z is defined as,

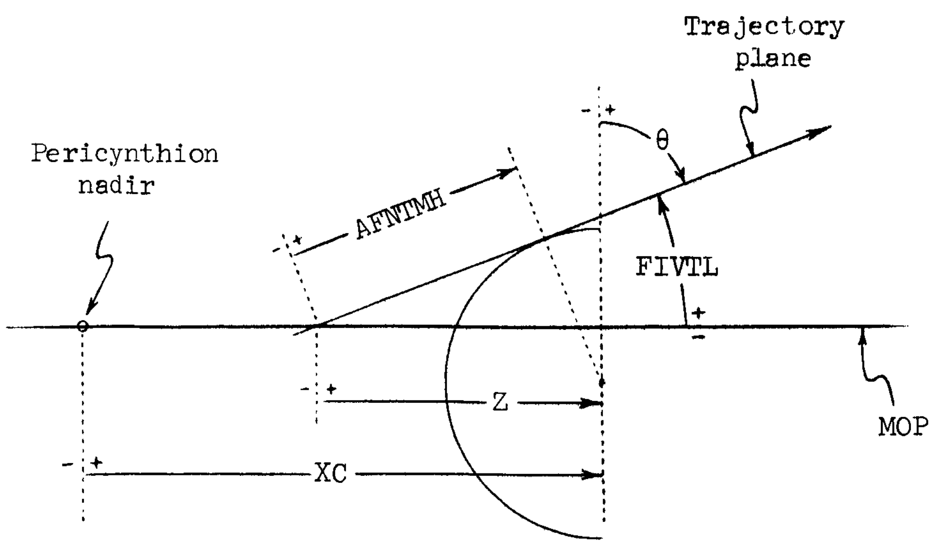
$$Z = -\ SIGN\ (1.0,\ S)\ \tan^{-1}\left(\frac{1}{\sqrt{\frac{1}{S^2} - 1}}\right)$$

However, if the condition $|FIVTL| < |YT|$ was originally detected, and YT was redefined so that tangency can be achieved, the above equations are not used to calculate Z. In these cases, Z is defined as equal to $\pi/2$ if FIVTL and YT have the same sign, or equal to $-\pi/2$ if FIVTL and YT have dissimilar signs. This insures that the trajectory node on the MOP, from which BETA and Z are measured, is the node nearest perigee.

With Z defined, BETA is simply calculated in radians as

$$BETA = Z - XC$$

The angle AFNTMH is then calculated using the intermediate variables COTH and SITH, which are the sine and cosine of the angle θ shown in the following figure.



$$COTH = \sin FIVTL \cos Z$$

$$SITH = \sqrt{1 - COTH^2}$$

The equation used to calculate AFNTMH is based upon plane trigonometry approximations, not pure spherical trigonometry.

$$AFNTMH = \frac{Z}{\cos FIVTL} + \frac{YC - YT}{\tan \theta}$$

Return is then executed.

2.4.4.2 Derivations or references: See references 1 and 3.

2.4.5 Restrictions.-

2.4.5.1 Range of numbers that can be processed: The range has not been determined.

2.4.5.2 Range of applicability: Subroutine SUBB is a specialized routine for use only with the CIST package.

2.4.5.3 Other programs required: No other programs are required.

2.4.6 Accuracy.- Accuracy is, in one sense, determined by the system FORTRAN library functions. The routine was designed for use in simulating translunar trajectories of the type used in nominal Apollo missions.

2.4.7 Coding information.- All calculations are in single precision.

2.4.8 Listing.-

```
      SUBROUTINE SUBB (IPERT,BETA,AFNTMH)
      COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
      EQUIVALENCE (PRE(13),XPC)
      EQUIVALENCE (PRE(14),YPC)
      EQUIVALENCE (XMS(5),AM)
      EQUIVALENCE (XMS(25),DS)
      EQUIVALENCE (TAR(5),FIVTL)
C
C                                    IN THIS VERSION OF SUBB, THE EFFECT
C                                    OF SOLAR GRAVITATION CANNOT BE
C                                    CONSIDERED SEPARATELY FROM THE
C                                    EFFECT OF EARTH OBLATENESS.
C
```

(Listing continued on next page)

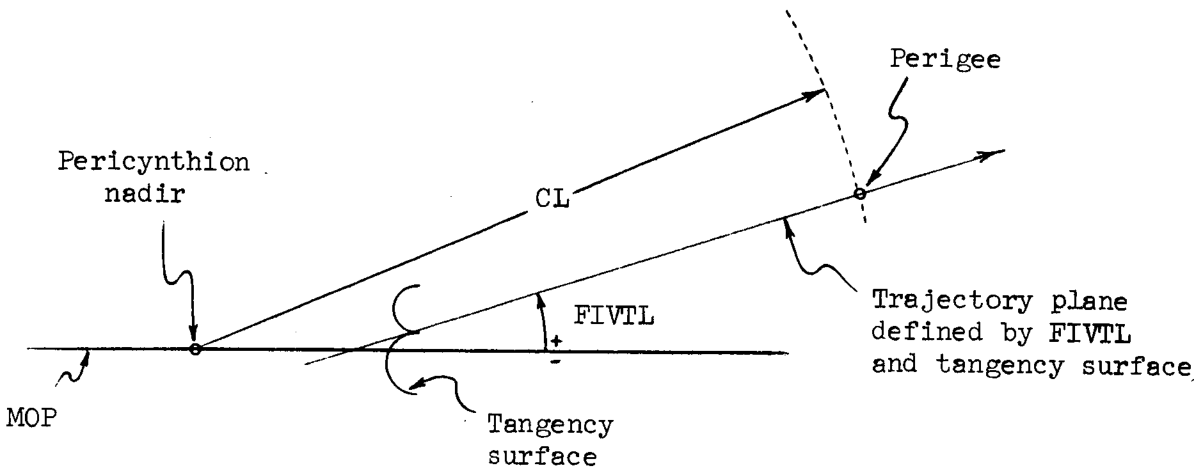
```
      DPR=57.29578
      XC=(1.93-0.037*XPC)/DPR
      YC=(-SIGN(1.0,FIVTL)*(0.60-0.02*XPC)-0.035*YPC)/DPR
      PLUS=0.0
      IF(IPERT.EQ.0) GO TO 20
      IF(IPERT.GE.2) GO TO 10
      PLUS=0.0317*COS(AM-18.0/DPR)
      GO TO 20
   10 PLUS=(0.0285+0.0115*COS(3.85*DS))*COS(AM-(10.0+5.0*COS(3.85*DS))
     1 /DPR)
   20 YT=(PLUS-YPC*(0.018165-0.000201*XPC))/DPR
      IF(ABS(YT).LT.ABS(FIVTL)) GO TO 30
      YT=SIGN(ABS(FIVTL),YT)
      S=-1.0
      IF(FIVTL*YT.LT.0.0) S=1.0
      Z=-SIGN(1.0,S)*90.0/DPR
      WRITE (6,900)
      GO TO 40
   30 S=(-SIN(YT-YC)-COS(FIVTL)*SIN(YC))/(SIN(FIVTL)*COS(YC))
      Z=-SIGN(1.0,S)*ATAN(1.0/SQRT(1.0/(S*S)-1.0))
   40 BETA=Z-XC
      COTH=SIN(FIVTL)*COS(Z)
      SITH=SQRT(1.0-COTH*COTH)
      AFNTMH=Z/COS(FIVTL)+(YC-YT)*COTH/SITH
      RETURN
  900 FORMAT(/30X,71HWARNING, TANGENCY SURFACE PROBLEM.  YT REDEFINED A
     1S SIGN(ABS(FIVTL),YT)//)
      END
```

2.5 Subroutine SUBCL

2.5.1 Identification.-

SUBCL (Trajectory Simulation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.5.2 Purpose.- In the present version of CIST, the position of the perigee of the simulated trajectory in the trajectory plane is defined by the angle CL. This angle is measured from pericynthion nadir to perigee, as shown in the following figure. CL is calculated in this subroutine.



In the reference, wherein the simulation of trajectories is described, perigee position is defined as the intersection of the trajectory plane with an empirical mechanism, the translunar perigee surface. This mechanism is not formulated in the present version of CIST.

2.5.3 Usage.-

2.5.3.1 Calling sequence: CALL SUBCL (IPERT, CL).

2.5.3.2 Arguments:

Parameter name	In/Out	Type	Description
IPERT	In	Integer	Perturbation instruction index =0, no perturbations considered =1, only earth oblateness considered =2, only solar gravitation considered =3, both earth oblateness and solar gravitation considered
CL	Out	Real	Angle between the pericynthion nadir and translunar perigee, in radians

2.5.3.3 Label common: All variables are input

Location in SIMUL block	MNEMONIC	Description
13	XPC	Longitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, measured counterclockwise from the extension of the earth-moon axis on the back side of the moon, in degrees.
14	YPC	Latitude of pericynthion of the simulated trajectory in a moon-centered MOP coordinate system, in degrees.
15	RPC	Radius of pericynthion of the simulated trajectory relative to the center of the moon, in nautical miles.
31	EMR	Earth-moon radius at the time of trajectory pericynthion, in nautical miles.
32	EMRDOT	$\dot{EMR}$, the first derivative of earth-moon radius at the time of trajectory pericynthion with respect to time, in knots.
36	AM	Argument of the moon in the MOP past its ascending node on the earth's equator, in radians.
51	SMOPL	Longitude of the sun in an earth-centered MOP coordinate system, at the time of trajectory pericynthion, measured counterclockwise from the position of the moon, in radians.
61	FIVTL	Inclination of the trajectory plane at perigee to the MOP, in radians. FIVTL is defined as negative if the trajectory is going below the MOP.

2.5.3.4 Sample usage: Refer to "Calling Sequence" and "Label Common" above.

2.5.3.5 Storage required: Coding occupies $273_8 (187_{10})$ locations.

2.5.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

2.5.4 Method.-

2.5.4.1 Statement of algorithms: The effects of solar gravitation (SECL) and earth oblateness (OBCL) upon the angle CL are first calculated.

$$SECL = \left(0.11 - (8 \times 10^{-7})\ EMR\right) \sin\ (2\ SMOPL + 0.611) - 0.01$$

$$OBCL = (1.6457056 \times 10^{-2}) \sin\ (2\ AM + 0.166)$$

Immediately after each calculation, the perturbation instruction index (IPERT) is tested to see if the given perturbation is to be considered in the calculation of CL. If it is not to be considered, SECL or OBCL is set equal to zero.

CL is then calculated by a lengthy empirical equation and converted to radians for final output.

$$\begin{aligned} CL = {} & 6.3814106 - (1.1030239 \times 10^{-5})\ EMR + (5.3567533 \times 10^{-3})\ EMRDOT \\ & + 6950./(RPC + 652.) + 0.0114\ |YPC| + 0.01892\ (YPC)(FIVTL) \\ & - \cos\ (FIVTL)(0.055\ XPC - 1.35) - \left[0.2457 - (4.5 \times 10^{-7})\ EMR\right]^2 \\ & \Big[888.88889 + 1.0/[5.0625\ XPC/(9.828 \times 10^5 - 1.8\ EMR) \\ & - 0.001125]\Big] + 0.055\ XPC + SECL + OBCL \end{aligned}$$

$$CL = CL/57.29578$$

2.5.4.2 Derivations or references: See reference 1.

2.5.5 Restrictions.-

2.5.5.1 Range of numbers that can be processed: This subroutine is designed for use in simulating translunar trajectories of the type used in nominal Apollo missions.

2.5.5.2 Range of applicability: Subroutine SUBCL is a specialized routine for use only with the CIST package.

2.5.5.3 Other programs required: No other programs are required.

2.5.6 Accuracy.- See " Restrictions" above.

2.5.7 Coding information.- All calculations and input/output are in single precision.

2.5.8 Listing.-

```
      SUBROUTINE SUBCL (IPERT,CL)
      COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
      EQUIVALENCE (PRE(13),XPC)
      EQUIVALENCE (PRE(14),YPC)
      EQUIVALENCE (PRE(15),RPC)
      EQUIVALENCE (XMS(1),EMR)
      EQUIVALENCE (XMS(2),EMRDOT)
      EQUIVALENCE (XMS(6),AM)
      EQUIVALENCE (XMS(21),SMOPL)
      EQUIVALENCE (TAR(6),FIVTL)
      DPR=57.295780
C
C     SECL IS THE EFFECT OF SOLAR GRAVITATION ON CL
C
      SECL=(0.11-8.0E-07*EMR)*SIN(2.0*SMOPL+0.611)-0.01
      IF(IPERT.LE.1) SECL=0.0
C
C     OBCL IS THE EFFECT OF EARTH OBLATENESS ON CL
C
      OBCL=1.6457066E-02*SIN(2.0*AM+0.166)
      IF(IPERT.EQ.0.OR.IPERT.EQ.2) OBCL=0.0
      CL=6.3814106-EMR*1.1030239E-05+EMRDOT*5.3567533E-03
     1 +6950.0/(RPC+652.0)+0.0114*ABS(YPC)+YPC*FIVTL*0.01892
     2 -COS(FIVTL)*(0.055*XPC-1.35)-(0.2457-EMR*4.5E-07)**2
     3 *(658.38889+1.0/(XPC*5.0625/(9.828E+05-EMR*1.8)-0.001125))
     4 +0.055*XPC+SECL+OBCL
      CL=CL/DPR
      RETURN
      END
```

2.6 Subroutine TLIMP

2.6.1 Identification.-

TLIMP (Empirical TLI Simulation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.6.2 Purpose.- The primary function of subroutine TLIMP is to calculate variables describing an optimum coplanar TLI thrust maneuver. These calculations are based upon a method of empirically simulating optimum TLI maneuvers which is described in the reference.

The secondary function of TLIMP is to define the osculating elements of the trajectory at perigee.

2.6.3 Usage.-

2.6.3.1 Calling sequence: CALL TLIMP.

2.6.3.2 Arguments: There are no arguments.

2.6.3.3. Label common:

Block name	Input	Output
SIMUL	7, 29, and 66	81-92

Refer to METHOD, Statement of algorithms:, for definition of the above parameters.

2.6.3.4 Sample usage: Refer to "Calling Sequence" and "Label Common" above.

2.6.3.5 Storage required: Coding occupies 520_8 (336_{10}) locations.

2.6.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

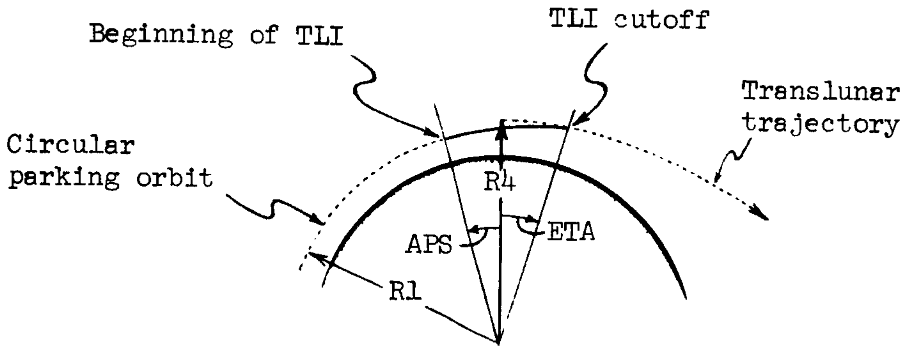
2.6.4 Method.-

2.6.4.1 Statement of algorithms: Input to and output from TLIMP consists of the following variables in the SIMUL common block.

Location in common block SIMUL	MNEMONIC	Definition
7	R1	Orbit radius, in nautical miles (input)
29	TTW	Thrust-to-weight ratio at beginning of TLI (input)
66	W	Trajectory energy in (international ft/sec)^2 (input) $W = C3/2 = \frac{V^2}{2} - \frac{u}{r}$
81	ETA	True anomaly of TLI cutoff, in radians
82	APS	Angle from beginning of coplanar TLI to perigee, in radians (equals α plus σ)
83	G7D	Flight-path angle at TLI cutoff, in degrees
84	DV	Characteristic velocity of coplanar TLI maneuver, in international ft/sec
85	FTOCO	Flight time on trajectory of TLI cutoff past perigee, in hours
86	R7	Radius of TLI cutoff, in nautical miles
87	E	Eccentricity of trajectory at perigee
88	R4	Perigee radius
89	P	Semilatus rectum
90	A	Semimajor axis
91	B	Semiminor axis
		(88–91: Conic elements of trajectory at perigee, in nautical miles)
92	COEF	Coefficient of flight time equations. Units are hr/n. mi.

Subroutine TLIMP defines a simulation of an optimum coplanar TLI thrust maneuver. The type of simulation used is described in reference 2. This type of TLI simulation, which uses relatively short empirical equations, is more accurate than simulations using multiple sets of lengthy polynomials.

The empirical equations of the simulation define ETA, APS, R4, and DV as functions of R1, W, and TTW.



The empirical equations used in TLIMP are those presented in reference 2. These equations use variables having metric units. These metric variables, RORB, UE, C3, and CDIF, are defined before the empirical equations are used.

FPKM Conversion factor, international ft/km

$$FPKM = 3280.8399$$

FKMPNM Conversion factor, km/ n. mi.

$$FKMPNM = 1.8520$$

RORB Radius of circular earth parking orbit, in kilometers

$$RORB = R1 \times FKMPNM$$

UE Gravitational constant of the earth, in km^3/sec^2

$$UE = 398603.20$$

C3 Trajectory energy, in $(km/sec)^2$

$$C3 = 2 \times W/FPKM^2$$

CDIF Difference between the energies of the circular parking orbit and the trajectory, in $(km/sec)^2$

$$CDIF = UE/RORB + C3$$

The four empirical equations are as follows:

$$\mathrm{ETA} = \tan^{-1}\left[\frac{\left(2.1397405 - \frac{\mathrm{RORB}}{5750.0}\right)\left(\frac{1.0143460}{\mathrm{TTW}} - 0.020\right)}{\frac{260.73592}{\mathrm{CDIF}} + 1.6338479}\right]$$

$$\mathrm{APS} = \tan^{-1}\left[\frac{\left(2.4185082 - \frac{\mathrm{RORB}}{4620.0}\right)\left(\frac{1.0365969}{\mathrm{TTW}} - 0.051020408\right)}{\frac{254.80898}{\mathrm{CDIF}} + 2.1846192}\right]$$

$$\mathrm{R4} = \mathrm{R1} + (-0.15664127\,\mathrm{CDIF}^3 + 34.568940\ \mathrm{CDIF}^2$$

$$-253.71417\ \mathrm{CDIF})/(\mathrm{RORB\ TTW}^2\ \mathrm{FKMPNM})$$

$$\mathrm{DV} = \left[\sqrt{\mathrm{CDIF} + \frac{\mathrm{UE}}{\mathrm{RORB}}} - \sqrt{\frac{\mathrm{UE}}{\mathrm{RORB}}} + \left[(\mathrm{DC} - 11.61)^2 \left(2.7022098 - \frac{\mathrm{RORB}}{3850.0}\right)\left(\frac{2.0264543 \cdot 10^{-6}}{\mathrm{TTW}} + (3.6327648 \times 10^{-8})\right)\right] \Big/ \mathrm{TTW}\right] \mathrm{FPKM}$$

After the empirical equations are used to define ETA, APS, R4, and DV, the conic variables are calculated for output in the SIMUL block. In doing this, four variables are defined which are not stored in SIMUL. These four variables are as follows:

VPGSQ	Velocity of perigee, squared
VPG	Velocity at perigee
HSQ	Angular momentum of vehicle, squared
H	Angular momentum of vehicle

The remaining algorithm of subroutine TLIMP is as follows:

$$\mathrm{VPGSQ} = 2\left(\mathrm{W} + \frac{\mathrm{U}}{\mathrm{R4}}\right)$$

$$\mathrm{VPG} = \sqrt{\mathrm{VPGSQ}}$$

$$\mathrm{HSQ} = (\mathrm{R4}^2)(\mathrm{VPGSQ})$$

$$\mathrm{H} = (\mathrm{R4})(\mathrm{VPG})$$

$$\mathrm{P} = \frac{\mathrm{HSQ}}{\mathrm{U}}$$

$$A = \frac{-U}{2W}$$

$$E = 1 - \frac{R4}{A}$$

$$B = \sqrt{|(A)\ (P)|}$$

$$R7 = \frac{P}{1 + E \cos ETA}$$

$$G8D = \tan^{-1}\left(\frac{(E)(R7)}{P} \sin ETA\right) 57.29578$$

$$COEF = \frac{6076.11549}{3600.0} \quad \frac{A}{H}$$

$$FTOCO = COEF \left(2B \tan^{-1}\left(\frac{R4}{B} \tan \frac{ETA}{2}\right) - (E)(R7) \sin ETA \right)$$

2.6.4.2 Derivations or references: See reference 2.

2.6.5. Restrictions.-

2.6.5.1 Range of numbers that can be processed: Input data should be restricted to the ranges of the data from which the simulation equations were derived. Parking orbit radius (R1) should be within about 40 n. mi. of 3550 n. mi., and thrust-to-weight ratio (TTW) should be between the approximate limits of .63 and .80. Trajectory energy can have any value from that of the parking orbit up to 0 (parabolic).

Violation of these limits will only result in questionable accuracy of the variables which are output by TLIMP.

2.6.5.2 Range of applicability: Subroutine TLIMP is a special routine intended primarily for use in the CIST package. It can be applied elsewhere providing all input falls within the definitions as defined herein.

2.6.5.3 Other programs required: There are no other program requirements.

2.6.6 Accuracy.- Detailed descriptions of the accuracy of the empirical equations used in TLIMP, to simulate optimum coplanar TLI thrust maneuver, can be found in references 5 and 6.

2.6.7 Coding information.- All input, output and computations are in single precision.

2.6.8 Listing.-

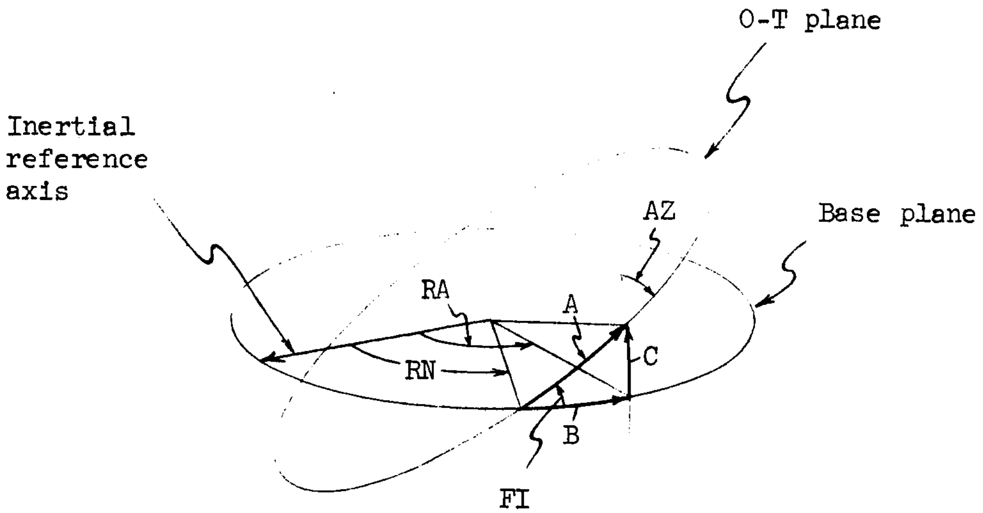
```
      SUBROUTINE TLIMP
      COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
      EQUIVALENCE (PRE(7),R1)
      EQUIVALENCE (PRE(29),TTW)
      EQUIVALENCE (TAR(11),W)
      EQUIVALENCE (TRAJ(1),ETA)
      EQUIVALENCE (TRAJ(2),APS)
      EQUIVALENCE (TRAJ(3),G7D)
      EQUIVALENCE (TRAJ(4),DV)
      EQUIVALENCE (TRAJ(5),FTOCO)
      EQUIVALENCE (TRAJ(6),R7)
      EQUIVALENCE (TRAJ(7),E)
      EQUIVALENCE (TRAJ(8),R4)
      EQUIVALENCE (TRAJ(9),P)
      EQUIVALENCE (TRAJ(10),A)
      EQUIVALENCE (TRAJ(11),B)
      EQUIVALENCE (TRAJ(12),COEF)
      J=.231670040E+13
      UE=398603.20
      FKMPNM=1.8520
      FPKM=3280.8399
      RORB=R1*FKMPNM
      C3=2.0*W/(FPKM*FPKM)
      CDIF=UE/RORB+C3
      ETA=ATAN((2.1397405-RORB/5750.0)*(1.0143460/TTW-0.020)/
     1 (260.73592/CDIF+1.6338479))
      APS=ATAN((2.4185082-RORB/4620.0)*(1.0365969/TTW-0.051020408)/
     1 (254.80898/CDIF+2.1846192))
      R4=R1+(-0.15664127*CDIF**3+34.568940*CDIF**2-253.71417*CDIF)/
     1 (RORB*TTW*TTW*FKMPNM)
      DV=(SQRT(CDIF+UE/RORB)-SQRT(UE/RORB)+((CDIF-11.61)**2*
     1 (2.7022098-RORB/3850.0)*(2.0264543E-06/TTW+3.6327648E-08))/
     2 TTW)*FPKM
      VPGSQ=2.0*(W+J/R4)
      VPG=SQRT(VPGSQ)
      HSQ=R4*R4*VPGSQ
      H=R4*VPG
      P=HSQ/J
      A=-J/(2.0*W)
      E=1.0-R4/A
      B=SQRT(ABS(A*P))
      R7=P/(1.0+E*COS(ETA))
      G7D=ATAN(E*R7/P*SIN(ETA))*57.29578
      COEF=6076.11549/3600.0*A/H
      FTOCO=COEF*(2.0*B*ATAN(R4/B*SIN(ETA/2.0)/COS(ETA/2.0))-
     1 E*R7*SIN(ETA))
      RETURN
      END
```

2.7 Subroutine GEOARG

2.7.1 Identification.-

GEOARG (Trigonometric Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.7.2 Purpose.- Subroutine GEOARG is used to calculate angles describing a state vector in an orbit-trajectory (O-T) plane in a polar coordinate system. It is not mandatory that the base plane of this coordinate system be the earth's equatorial plane although this is usually the case.



Given FI, A, and RN, GEOARG calculates AZ, B, C, and RA. All parameters, both input and output, are angles, the units of which are radians.

2.7.3 Usage.-

2.7.3.1 Calling sequence: CALL GEOARG (FI, A, RN, AZ, B, C, RA)

2.7.3.2 Arguments:

Parameter name	In/Out	Type	Description
FI	In	Real	Inclination of orbit-trajectory plane to base plane of polar coordinate system. The present version of GEOARG is limited to posigrade conditions between the O-T and base planes. This limitation necessitates the following restriction of FI. In radians $0 \leq FI \leq \pi/2$
A	In	Real	Argument of state vector in O-T plane past the ascending node of this plane on the base plane of the polar coordinate system. In radians $-\pi < A \leq \pi$
RN	In	Real	Longitude or right ascension in the base plane of the ascending node of the O-T plane measured relative to some inertial reference axis. If the base plane is the earth's equatorial plane, RN would be the right ascension of the ascending node of the O-T plane relative to the first point of Aries. In radians $-\pi < RN \leq \pi$
AZ	Out	Real	Azimuth of the O-T plane at the state vector in the polar coordinate system. In radians $0 \leq AZ \leq \pi$
B	Out	Real	Longitude of the state vector, measured in the base plane, past the ascending node of the O-T plane on the base plane. In radians $-\pi < B \leq \pi$
C	Out	Real	Declination of state vector relative to base plane of coordinate system. In radians $-\pi/2 \leq C \leq \pi/2$

Parameter name	In/Out	Type	Description
RA	Out	Real	Longitude or right ascension of the state vector, measured in the base plane from the inertial reference axis. If the base plane is the earth's equatorial plane, RA would be conventional right ascension measured from the first point of Aries. In radians $-\pi < RA \le \pi$

2.7.3.3 Label common: There is no label common.

2.7.3.4 Sample usage: Refer to "Calling Sequence" above.

2.7.3.5 Storage required: Coding occupies 416_8 (270_{10}) locations.

2.7.3.6 Error Codes and diagnostics: There are no error codes or diagnostics.

2.7.4 Method.-

2.7.4.1 Statement of algorithms: Initially, tests are made for special situations wherein the calculations of AZ, B, and C do not require the normally used arctangent equations.

If $FI = 0$,	$AZ = \pi/2$	$B = A$	$C = 0$
If $FI = \pi$,	$AZ = -\pi/2$	$B = -A$	$C = 0$
If $FI = \pi/2$ and $\lvert A\rvert < \frac{\pi}{2}$,	$AZ = 0$	$B = 0$	$C = A$
If $FI = \frac{\pi}{2}$ and $A \ge \frac{\pi}{2}$,	$AZ = \pi$	$B = \pi$	$C = \pi - A$
If $FI = \frac{\pi}{2}$ and $A \le -\frac{\pi}{2}$,	$AZ = \pi$	$B = \pi$	$C = -A - \pi$
If $A = \pi/2$,	$AZ = \pi/2$	$B = \pi/2$	$C = FI$
If $A = -\pi/2$,	$AZ = \pi/2$	$B = -\pi/2$	$C = -FI$
If $A = 0$,	$AZ = \pi/2 - FI$	$B = 0$	$C = 0$
If $A = \pi$,	$AZ = \pi/2 + FI$	$B = \pi$	$C = 0$

If none of the above conditions exist, AZ, B, and C are calculated by the following arctangent equations. A two quadrant (ATAN) function is used to calculate C. A four quadrant (ATAN2) function is used to calculate AZ and B, the arguments being chosen to provide answers in the proper quadrant.

$$AZ = \tan^{-1}\left(\frac{\text{ctn FI}}{\cos A}\right)$$

$$B = \tan^{-1}(\tan A \cos FI)$$

$$C = \tan^{-1}(\tan FI \sin B)$$

In all cases, RA is calculated as the sum of RN and B, defined between $\pm\pi$ by calling subroutine HELP.

2.7.4.2 Derivations or references: There are no derivations or references.

2.7.5. Restrictions.-

2.7.5.1 Range of numbers that can processed: See the range restrictions of input parameters under argument description.

2.7.5.2 Range of applicability: GEOARG is designed for use in the CIST system. However, it may be applied to other programs within the confines of its definition.

2.7.5.3 Other programs required: Subroutine HELP is required.

2.7.6 Accuracy.- Limits of accuracy are determined by the limits of the system FORTRAN library functions.

2.7.7 Coding information.- All calculations and input and output are in single precision.

2.7.8 Listing.-

```
      SUBROUTINE GEOARG (FI,A,RN,AZ,B,C,RA)
C
C  INPUT = FI = INCLINATION OF O-T PLANE TO BASE PLANE
C          A  = ARGUMENT PAST ASCENDING NODE IN O-T PLANE
C          RN = RIGHT ASCENSION OF ASCENDING NODE
C
C OUTPUT = AZ = AZIMUTH
C          B  = LONGITUDE IN BASE PLAE PAST ASCENDING NODE
```

(Listing continued on next page)

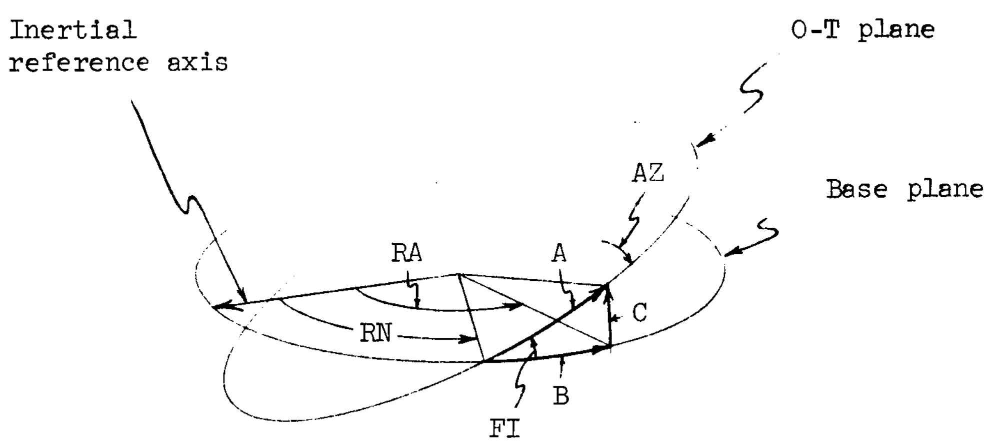
```
C            C  = DECLINATION RELATIVE TO BASE PLANE
C            RA = RIGHT ASCENSION
C
C      ALL ANGLES ARE IN RADIANS
C
       PI=3.1415927
       HP=1.57079635
       IF(FI.NE.0.0) GO TO 10
       AZ=HP
       B=A
       C=0.0
       GO TO 70
 10    IF(FI.NE.PI) GO TO 20
       AZ=-HP
       B=-A
       C=0.0
       GO TO 70
 20    IF(FI.NE.HP) GO TO 30
       AZ=0.0
       B=0.0
       C=A
       IF(ABS(A).LT.HP) GO TO 70
       AZ=PI
       B=PI
       C=SIGN(PI,A)-A
       GO TO 70
 30    IF(ABS(A).NE.HP) GO TO 40
       AZ=SIGN(HP,HP-FI)
       B=SIGN(HP,A*(HP-FI))
       C=SIGN(FI,A)
       GO TO 70
 40    IF(A.NE.0.0) GO TO 50
       B=0.0
       C=0.0
       AZ=HP-FI
       GO TO 70
 50    IF(A.NE.PI) GO TO 60
       B=PI
       C=0.0
       AZ=HP+FI
       CALL HELP (AZ)
       GO TO 70
 60    AZ=ATAN2(COS(FI),SIN(FI)*COS(A))
       B=ATAN2(SIN(A)*COS(FI),COS(A))
       C=ATAN(SIN(FI)*SIN(B)/COS(FI))
 70    RA=RN+B
       CALL HELP (RA)
       RETURN
       END
```

2.8 Subroutine GEOLAT

2.8.1 Identification.-

GEOLAT (Trigonometric Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.8.2 Purpose.- Subroutine GEOLAT is used to calculate angles describing a state vector in an orbit-trajectory (O-T) plane in a polar coordinate system. It is not mandatory that the base plane of this coordinate system be the earth's equatorial plane, although this is usually the case.



Given FI, C, and RA, GEOLAT calculates A, B, AZ, and RN. There are two solutions to this type of problem, one having azimuth greater than $\pi/2$, the other having azimuth less than $\pi/2$. The desired solution is indicated by the index I, which is input in the calling argument. With the exception of I, all parameters are angles, the units of which are radians.

2.8.3 Usage.-

2.8.3.1 Calling sequence: CALL GEOLAT (FI, C, RA, I, A, B, AZ, RN).

2.8.3.2 Arguments:

Parameter name	In/Out	Type	Description
FI	In	Real	Inclination of O-T plane to the base plane of the polar coordinate system. The present coding of GEOLAT is limited to posigrade conditions between the O-T and base planes. This limitation necessitates the following restriction of FI. In radians $0 \leq FI \leq \pi/2$
C	In	Real	Declination of the state vector relative to the base plane of the coordinate system. In radians $-\pi/2 \leq C \leq \pi/2$
RA	In		Longitude or right ascension of the state vector, measured in the base plane from the inertial reference axis. If the base plane is the earth's equatorial plane, RA would be conventional right ascension measured from the first point of Aries. In radians $-\pi < RA \leq \pi$
I	In	Integer	Index defining which of two possible solutions is desired. I = 1 solution with $0 \leq AZ \leq \pi/2$ I = 2 solution with $\pi/2 < AZ \leq \pi$
A	Out	Real	Argument of state vector in the O-T plane past the ascending node of this plane on the base plane of the polar coordinate system. In radians $-\pi < A \leq \pi$
B	Out	Real	Longitude of the state vector, measured in the base plane, past the ascending node of the O-T plane on the base plane. In radians $-\pi < B \leq \pi$

Parameter name	In/Out	Type	Description
AZ	Out	Real	Azimuth of the O-T plane at the state vector in the polar coordinate system, in radians. Since the present coding of GEOLAT is limited to posigrade cases, $0 \leq AZ \leq \pi$
RN	Out	Real	Longitude or right ascension in the base plane of the ascending node of the O-T plane measured relative to some inertial reference axis. If the base plane is the earth's equatorial plane. RN would be the right ascension of the ascending node of the O-T plane relative to the first point of Aries. In radians

2.8.3.3 Label common: There is no label common.

2.8.3.4 Sample usage: Refer to "Calling Sequence " above.

2.8.3.5 Storage required: Coding occupies 342_8 (226_{10}) locations.

2.8.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

2.8.4 Method.-

2.8.4.1 Statement of algorithms: Initially, tests are made for values of C being equal to 0, FI, and -FI. In these special cases the calculation of A, B, and AZ are very straightforward and the arctangent equations normally used are bypassed.

IF C = FI, A = $\pi/2$, B = $\pi/2$, AZ = $\pi/2$.

If C = -FI, A = $-\pi/2$, B = $-\pi/2$, AZ = $\pi/2$,

If C = 0 and I = 1, A = 0, B = 0, AZ = $\pi/2$ - FI.

If C = 0 and I = 2, A = π, B = π, AZ = $\pi/2$ + FI.

When GEOLAT was first coded, there was no arcsine function available in the function library. Consequently, A is calculated using an arctangent function, the argument of which is the expression for tangent as a function of cosecant. The cosecant of A is equal to sin (FI)/sin (C), and is given in the address CSCA. The expression for the tangent of A is in double precision to achieve better accuracy near A = $\pm\ \pi/2$, where the slope of the cosecant curve is near zero. The value of A given by the ATAN function will be in the first quadrant. Proper quadrant allocation is achieved by subsequent statements testing I and C.

With A defined in the proper quadrant, B and AZ are calculated using a four quadrant arctangent function (ATAN2), the arguments of which are designed to provide answers in the proper quadrant.

$$B = \tan^{-1} (\tan A \cos FI) \cdot$$

$$AZ = \tan^{-1}\left[\frac{\text{ctn FI}}{\cos A}\right] \cdot$$

In all cases, RN is calculated as RA - B, defined between $-\pi$ and π by calling subroutine HELP.

2.8.4.2 Derivations or references: There are no derivations or references.

2.8.5 Restrictions.-

2.8.5.1 Range of numbers that can be processed: Refer to the limits outlined in the section "Arguments".

2.8.5.2 Range of applicability: Subroutine GEOLAT is a routine designed within the framework of the CIST system. Its use in other programs must lie within the limits of its definition.

2.8.5.3 Other programs required: Subroutine HELP is required.

2.8.6 Accuracy.- Accuracy is determined in general by the limits of the system FORTRAN library functions.

2.8.7 Coding information.- All calculations and input and output are in single precision.

2.8.8 Listing.-

```
      SUBROUTINE GEOLAT (FI,C,RA,I,A,B,AZ,RN)
C
C   INPUT = FI = INCLINATION
C           C  = DECLINATION
C           RA = RIGHT ASCENSION
C           I  = QUADRANT INSTRUCTION
C
C                FOR A AND B IN FIRST OR FOURTH QUADRANTS, I=1
C                FOR A AND B IN SECOND OR THIRD QUADRANTS, I=2
C
C  OUTPUT = A  = ARGUMENT PAST ASCENDING NODE
C           B  = LONGITUDE PAST ASCENDING NODE
C           AZ = AZIMUTH
C           RN = RIGHT ASCENSION OF ASCENDING NODE
C
C   ALL ANGLES ARE IN RADIANS
C
      DOUBLE PRECISION CSCA
      PI=3.1415927
      IF(ABS(C).NE.FI) GO TO 10
      AZ=PI/2.0
      A=SIGN(AZ,C)
      B=A
      GO TO 40
   10 IF(C.NE.0.0) GO TO 30
      IF(I.EQ.2) GO TO 20
      A=0.0
      B=0.0
      AZ=PI/2.0-FI
      GO TO 40
   20 A=PI
      B=PI
      AZ=PI/2.0+FI
      GO TO 40
   30 CSCA=SIN(FI)/SIN(C)
      A=ATAN(1.0/DSQRT(CSCA*CSCA-1.0D0))
      IF(I.EQ.2) A=PI-A
      IF(C.LT.0.0) A=-A
      B=ATAN2(SIN(A)*COS(FI),COS(A))
      AZ=ATAN2(COS(FI),SIN(FI)*COS(A))
   40 RN=RA-B
      CALL HELP (RN)
      RETURN
      END
```

2.9 Subroutine HELP

2.9.1 Identification.-

HELP (Quadrant Allocation Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

2.9.2 Purpose.- Subroutine HELP redefines by mod 2π a given angle (X) to a value within the interval $-\pi < X \leq \pi$.

2.9.3 Usage.-

2.9.3.1 Calling sequence: CALL HELP (X)

2.9.3.2 Arguments:

Parameter name	In/Out	Dimension	Type	Description
X	In	1	Real	Input angle, in radians, no limitation in sign or magnitude
X	Out	1	Real	Output angle, in radians, equal to the input angle defined in the interval $-\pi < X \leq \pi$ by mod 2π.

2.9.3.3 Label common: There is no label common.

2.9.3.4 Sample usage: Refer to "Calling Sequence" above.

2.9.3.5 Storage required: Coding occupies 67_8 (55_{10}) locations.

2.9.3.6 Error Codes and diagnostics: There are no error codes or diagnostics.

2.9.4 Method.-

2.9.4.1 Statement of algorithms:

If $X > \pi$, then $X = X - 2\pi$ until $X \leq \pi$

If $X \leq -\pi$, then $X = X + 2\pi$ until $X > -\pi$

2.9.4.2 Derivations or references: There are no derivations or references.

2.9.5 Restrictions.-

2.9.5.1 Range of numbers that can be processed: The only limitations imposed are those set by the hardware and software capability of the computer. The argument is single precision.

2.9.5.2 Range of applicability: Subroutine HELP is a general routine that can be used with any program.

2.9.5.3 Other programs required: No other programs are required.

2.9.6 Accuracy.-

2.9.6.1 Method of determination: See "Restrictions".

2.9.7 Coding information.- All calculations and input/output are in single precision.

2.9.8 Listing.-

```
      SUBROUTINE HELP (X)
C
C  THE ANGLE X, IN RADIANS, IS DEFINED BETWEEN +PI AND -PI
C
      PI=3.1415927
   10 IF(X.LE.PI) GO TO 20
      X=X-2.0*PI
      GO TO 10
   20 IF(X.GT.(-PI)) GO TO 30
      X=X+PI*2.0
      GO TO 20
   30 RETURN
      END
```

3.0 SUBROUTINE UPDATE

3.1 Identification

UPDATE (TLI targeting update routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

3.2 Purpose

Subroutine UPDATE modifies TLI targeting elements defined in subroutine CIST to compensate for a dispersion in the time of the parking orbit state vector. This time dispersion is the difference between the actual, or real time, time of the parking orbit state vector, and the time defined by CIST as being necessary for coplanar TLI.

The modification of TLI targeting elements performed in UPDATE is described in reference 7; it consists of a change in the inertial position of the unit $\hat{M}$ vector, with no changes in either the hypersurface radius (σ) or trajectory energy (C3 or W). The change in $\hat{M}$ position consists of a rotation through the angle ψ around the angular momentum vector of the moon as defined at the time of trajectory pericynthion.

In reference 7, several different methods are described for calculating the angle ψ. In this version of UPDATE, ψ is calculated by the following equation. (See USAGE for definitions of the variables).

$$\psi = \mathrm{WM}\left(\frac{\mathrm{WV} - (\mathrm{WE}\cos\mathrm{FIV})}{\mathrm{WV} - (\mathrm{WM}\cos\mathrm{FIVTL})}\right) \quad \mathrm{TOIDIS}$$

Subroutine UPDATE has been essentially unchanged since its original documentation in reference 8.

3.3 Usage

3.3.1 Calling sequence.- CALL UPDATE (TOIDIS)

Previous to calling UPDATE, two things must be done: (1) CIST must be called, thus defining the variables in the SIMUL block which are input to UPDATE, and (2) TOIDIS must be defined.

TOIDIS is the time dispersion in hours of the input state vector in earth parking orbit. There can be a great deal of confusion concerning the sign of TOIDIS. Hopefully, the following detailed definition of TOIDIS will eliminate or at least reduce this confusion.

$$\text{TOIDIS} = \begin{bmatrix}\text{Real time (actual time)} \\ \text{of input parking orbit} \\ \text{state vector, in hours} \\ \text{relative to a base time}\end{bmatrix} - \begin{bmatrix}\text{Time of input parking} \\ \text{orbit state vector de-} \\ \text{fined by CIST for co-} \\ \text{planar TLI, in hours} \\ \text{relative to a base time}\end{bmatrix}$$

Consequently, TOIDIS is positive if the actual orbit state vector is late for coplanar TLI, and negative if it is early.

3.3.2 Arguments.-

Parameter name	In/Out	Dimension	Type	Description
TOIDIS	In	1	Real	See above

3.3.3 Label common.- All variables are input with the exception of locations 76, 78, 79, and 80.

Location in SIMUL block	MNEMONIC	Description
44	WM	Angular velocity of the moon, in rad/hr
45	XHM	Cartesian components of the angular momentum vector of the moon, in e.r.^{2}/hr
46	YHM	
47	ZHM	
58	WV	Angular velocity of the vehicle in earth parking orbit, in rad/hr
59	WE	Angular velocity of the earth's rotation, in rad/hr
60	FIV	Inclination of the earth parking orbit to the earth's equator, in radians
61	FIVTL	Inclination of the trajectory plane at perigee to the MOP for coplanar TLI, in radians. FIVTL is defined as negative if the trajectory is going below the MOP.

Location in SIMUL block	MNEMONIC	Description
63	C3	Declination of the $\hat{M}$ target vector defined by CIST for coplanar TLI, in radians
64	RA3	Right ascension of the $\hat{M}$ target vector defined by CIST for coplanar TLI, in radians. $-\pi < RA3 \leq \pi$
67	XMH	Cartesian components of the original unit $\hat{M}$ TLI target vector as output by CIST for coplanar TLI.
68	YMH	
69	ZMH	
71	RNV	Right ascension of the ascending node of the earth parking orbit on the earth's equator, in radians. $-\pi < RNV \leq \pi$
74	A1	Argument in the earth parking orbit plane of the input state vector, past the ascending node on the earth's equator, in radians. $-\pi < A1 \leq \pi$
76	DEL	δ, the latitude of the updated unit $\hat{M}$ TLI targeting vector relative to the dispersed parking orbit plane, in radians.
78	XUMH	Cartesian components of the updated unit $\hat{M}$ TLI targeting vector.
79	YUMH	
80	ZUMH	

3.3.4 Sample usage.- Refer to "Calling Sequence" and Label Common" above.

3.3.5 Storage required.- Coding occupies $1021_8(529_{10})$ locations.

3.3.6 Error codes and diagnostics.- There are no error codes or diagnostics.

3.4 Method

3.4.1 Statement of algorithms.- The angle ψ is calculated in both radians (PSI) and degrees (PSID).

$$\text{PSI} = \text{WM}\left(\frac{\text{WV} - (\text{WE} \cos \text{FIV})}{\text{WV} - (\text{WM} \cos \text{FIVTL})}\right) \quad \text{TOIDIS}$$

$$\text{PSID} = \text{PSI}\ 57.29578$$

The original $\hat{M}$ target vector defined in CIST (denoted below as $\hat{M}_o$) is then rotated around the angular momentum vector of the moon (denoted below as $\vec{H}_m$) by the angle PSI to the updated position (denoted below as $\hat{M}_u$)

$$\hat{M}_u = \frac{\vec{H}_m \cdot \hat{M}_o}{|\vec{H}_m|^2}\vec{H}_m + \frac{\sin \text{PSI}}{|\vec{H}_m|}(\vec{H}_m \times \hat{M}_o) + \frac{\cos \text{PSI}}{|\vec{H}_m|^2}\left((\vec{H}_m \times \hat{M}_o) \times \vec{H}_m\right)$$

The angular momentum vector, $\vec{H}_v$, corresponding to the parking orbit state vector with the time dispersion is calculated. An intermediate variable, DISRNV, which is the right ascension of the ascending node of the parking orbit plane with time dispersion, is used in doing this.

$$\text{DISRNV} = \text{RNV} + (\text{WE})(\text{TOIDIS})$$

$$HV_x = \sin \text{FIV} \sin \text{DISRNV}$$

$$HV_y = -\sin \text{FIV} \cos \text{DISRNV}$$

$$HV_z = \cos \text{FIV}$$

The angle DEL is then calculated as $\pi/2$ less the angle from $\vec{H}_v$ to $\hat{M}_u$, in radians and degrees (DELD). For printout purposes, the right ascension and declinations of $\hat{M}_o$ and $\hat{M}_u$ are calculated.

For checkout purposes, the central angle CAR, in radians, (CA is in degrees) between $\hat{M}_o$ and $\hat{M}_u$ is calculated. This angle should be very close to PSI in both magnitude and sign.

Finally, the angle ARC in the parking orbit plane from the state vector with the time dispersion to the perpendicular plane containing $\vec{H}_v$ and $\hat{M}_u$ is calculated in radians and degrees (ARCD). In doing this, the position vector, $\overrightarrow{POV}$, corresponding to the time dispersion orbit state vector is defined.

CALL GEOARG (FIV, A1, DISRNV, AZ1, B1, C1, RA1)

$$POV_x = \cos C1 \cos RA1$$

$$POV_y = \cos C1 \sin RA1$$

$$POV_z = \sin C1$$

ARC is then calculated as the angle from $\overrightarrow{POV}$ to $(\vec{H}_v \times \hat{M}_u)$ less $\pi/2$. The hypothetical coast time, CT, in orbit over ARC is then calculated as ARC/WV.

It is important not to confuse ARC and CT with the coast arc and time in orbit from the input orbit state vector to the beginning of the TLI thrust maneuver. In order to calculate these values, it is necessary to know the value of the angle α, which is measured in the parking orbit plane from the beginning of the TLI maneuver to the perpendicular projection of $\hat{M}_u$.

The value of α calculated in TLIMP for CIST is for coplanar TLI only and is not applicable to plane change TLI's, which will occur whenever δ is not zero.

3.4.2 Derivations or references.- See references 7 and 8.

3.5 Restrictions

3.5.1 Range of numbers that can be processed.- This is to be determined.

3.5.2 Range of applicability.- Subroutine UPDATE is a specialized routine for use only with the CIST package.

3.5.3 Other programs required.- Subroutines ANGLE, CROSS, and GEOARG, HELP are also required.

3.6 Accuracy.- Accuracy is to be determined.

3.7 Coding information.- All calculations and input and output are in single precision.

3.8 Listing.-

```
SUBROUTINE UPDATE (TOIDIS)
COMMON / SIMUL / PRE(30),XMS(25),TAR(25),TRAJ(20)
DIMENSION POV(3),HV(5),DUM(5)
EQUIVALENCE (XMS(14),WM)
EQUIVALENCE (XMS(15),XHM)
EQUIVALENCE (XMS(16),YHM)
EQUIVALENCE (XMS(17),ZHM)
EQUIVALENCE (TAR(3),WV)
EQUIVALENCE (TAR(4),WE)
EQUIVALENCE (TAR(5),FIV)
EQUIVALENCE (TAR(6),FIVTL)
EQUIVALENCE (TAR(8),C3)
EQUIVALENCE (TAR(9),RA3)
EQUIVALENCE (TAR(12),XMH)
EQUIVALENCE (TAR(13),YMH)
EQUIVALENCE (TAR(14),ZMH)
EQUIVALENCE (TAR(16),RNV)
EQUIVALENCE (TAR(19),A1)
EQUIVALENCE (TAR(21),DEL)
EQUIVALENCE (TAR(23),XJMH)
EQUIVALENCE (TAR(24),YJMH)
EQUIVALENCE (TAR(25),ZJMH)
PI=3.1415927
DPR=57.29578
PSI=WM*TOIDIS*((WV-WE*COS(FIV))/(WV-WM*COS(FIVTL)))
PSID=PSI*DPR
DOT=XMH*XHM+YMH*YHM+ZMH*ZHM
HOMSQ=XHM*XHM+YHM*YHM+ZHM*ZHM
HOM=SQRT(HOMSQ)
CO1=DOT/HOMSQ
CO2=SIN(PSI)/HOM
CO3=COS(PSI)/HOMSQ
XC1=YHM*ZMH-ZHM*YMH
YC1=ZHM*XMH-XHM*ZMH
ZC1=XHM*YMH-YHM*XMH
```

(Listing continued on next page)

```
      XC2=YC1*ZHM-ZC1*YHM
      YC2=ZC1*XHM-XC1*ZHM
      ZC2=XC1*YHM-YC1*XHM
      XUMH=CO1*XHM+CO2*XC1+CO3*XC2
      YUMH=CO1*YHM+CO2*YC1+CO3*YC2
      ZUMH=CO1*ZHM+CO2*ZC1+CO3*ZC2
C
C   THE FOLLOWING STATEMENTS ARE FOR THE CALCULATION OF DELTA AND
C   OTHER PARAMETERS OF INTEREST
C
      DISRNV=RNV+WE*TOIDIS
      CALL HELP (DISRNV)
      HV(1)=SIN(FIV)*SIN(DISRNV)
      HV(2)=-SIN(FIV)*COS(DISRNV)
      HV(3)=COS(FIV)
      CALL CROSS (HV,XUMH,DUM)
      CALL ANGLE (HV,XUMH,DUM,DEL,DELD)
      DEL=PI/2.0-DEL
      DELD=DEL*DPR
      RA3D=RA3*DPR
      C3D=C3*DPR
      RA3U=ATAN2(YUMH,XUMH)*DPR
      C3U=ATAN(ZUMH/SQRT(XUMH*XUMH+YUMH*YUMH))*DPR
      CALL ANGLE (XMH,XUMH,XHM,CAR,CA)
      CALL GEOARG (FIV,A1,DISRNV,AZ1,B1,C1,RA1)
      POV(1)=COS(C1)*COS(RA1)
      POV(2)=COS(C1)*SIN(RA1)
      POV(3)=SIN(C1)
      CALL ANGLE (POV,DUM,HV,ARC,ARCD)
      ARC=ARC-PI/2.0
      CALL HELP (ARC)
      ARCD=ARC*DPR
      CT=ARC/WV
      WRITE (6,900) TOIDIS,PSID,C3D,RA3D,C3U,RA3U,CA,DELD,ARCD,CT
  900 FORMAT(1H1///51H      DISPERSION IN HOURS OF THE ORBIT STATE VECT
     1OR/18X,36HFROM THE VALUE PRESCRIBED BY CIST  =,F13.8,38X,
     224HDECL,13X,5HRYTAS//48X,6HPSI  =,F13.8,15X,13HORIGINAL MHAT,2F18.8
     33/82X,13HUPDATED  MHAT,2F18.8/
     4 9X,42HCENTRAL ANGLE BETWEEN MHAT DEFINED BY CIST/
     5 31X,23HAND THE UPDATED MHAT  =,F13.8//46X,8HDELTA  =,F13.8//
     6 51H ARC FROM ORBIT STATE VECTOR, WITH TIME DISPERSION,/
     7 7X,44HTO THE PLANE CONTAINING UPDATED MHAT AND THE/
     8 16X,38HANGULAR MOMENTUM VECTOR OF THE MOON  =,F13.8//
     951H   HYPOTHETICA COAST TIME, IN HOURS, OVER THE ABOVE/
     1 27X,27HARC IN THE PARKING ORBIT  =,F13.8)
      RETURN
      END
```

4.0 SUBROUTINES OF UPDATE

4.1 Subroutine ANGLE

4.1.1 Identification.-

ANGLE
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

4.1.2 Purpose.- Subroutine ANGLE calculates the angle between two 3-space vectors VEC1 and VEC2. This angle is defined between $-\pi$ and π. The proper quadrant of the angle is determined by the orientation of VEC1 and VEC2 with respect to a third 3-space vector VEC3.

4.1.3 Usage.-

4.1.3.1 Calling sequence: CALL ANGLE (VEC1, VEC2, VEC3, XR, XD)

4.1.3.2 Arguments:

Parameter name	In/Out	Dimension	Type	Description
VEC1	In	3	Real	Input vector
VEC2	In	3	Real	Input vector
VEC3	IN	3	Real	Input vector
XR	Out	1	Real	Angle between VEC1 and VEC2 in radians. $-\pi < XR \leq \pi$
XD	Out	1	Real	Angle between VEC1 and VEC2 in degrees. $-180° < XD \leq 180°$

4.1.3.3 Label common: There is no label common.

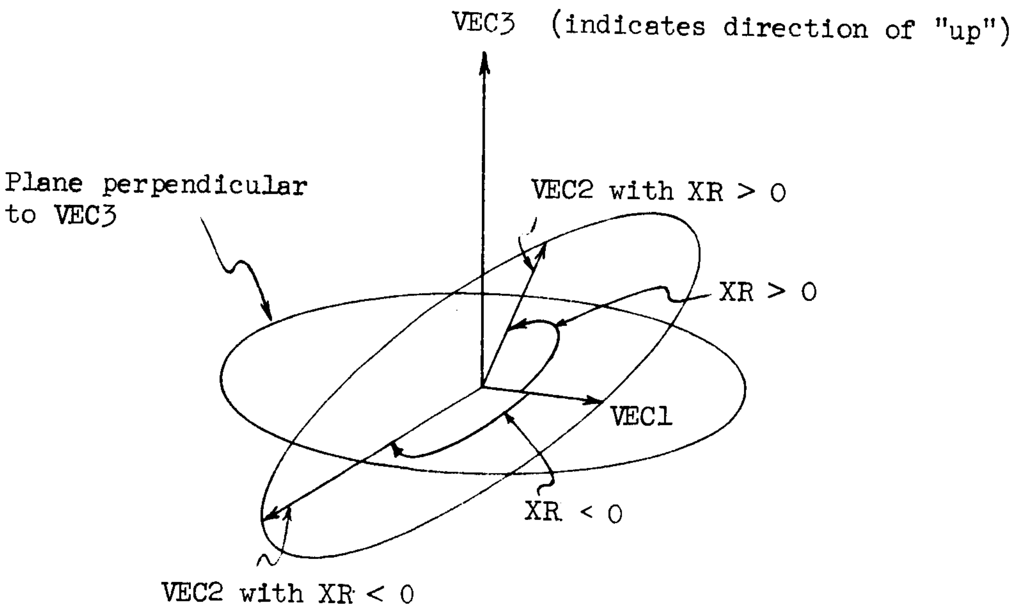
4.1.3.4 Sample usage: Refer to "Calling Sequence" above.

4.1.3.5 Storage required: Coding occupies $116_8 (78_{10})$ locations.

4.1.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

4.1.4 Method.-

4.1.4.1 Statement of algorithms: First, using a four quadrant arctangent function (ATAN2), the angle XR from VEC1 to VEC2 is calculated. The numerator of the ATAN2 argument is the magnitude of VEC1 × VEC2, the address of which is H(4). The denominator of the ATAN2 argument is the dot product of VEC1 and VEC2, the address of which is DUM. H(4) is always positive but DUM can have negative or positive values. Thus, the value of XR given directly by the ATAN2 function will always be in the first or second quadrants. This value of XR is not redefined if the angle between VEC1 × VEC2 and VEC3 is less or equal to $\pi/2$. This condition is tested by redefining DUM as the dot product of VEC1 × VEC2 and VEC3, and then testing the sign of DUM. If DUM is negative, indicating that the angle between VEC1 × VEC2 and VEC3 is between $\pi/2$ and π, XR is redefined as -XR. XD is then defined as the equivalent of XR in degrees. VEC3 can be considered as representing the direction of "up", and XR is the angle from VEC1 to VEC2, measured in a counterclockwise direction around VEC3.



4.1.4.2 Derivations or references: There are no references or derivations.

4.1.5 Restrictions.-

4.1.5.1 Range of numbers that can be processed: Any vectors represented in Cartesian coordinates can be processed.

4.1.5.2 Other programs required: CROSS, DOT are also required.

4.1.6 Accuracy.- The limits of accuracy will be imposed by the system FORTRAN library functions.

4.1.7 Coding information.- All computations and input and output are in single precision.

4.1.8 Listing.-

```
SUBROUTINE ANGLE (VEC1,VEC2,VEC3,XR,XD)
DIMENSION VEC1(3),VEC2(3),VEC3(3),H(5)
CALL CROSS (VEC1,VEC2,H)
CALL DOT (VEC1,VEC2,DUM)
XR=ATAN2(H(4),DUM)
CALL DOT (H,VEC3,DUM)
IF(DUM.LT.0.0) XR=-XR
XD=XR*57.29578
RETURN
END
```

4.2 Subroutine DOT

4.2.1 Identification.-

DOT (Vector Dot Product Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

4.2.2 Purpose.- Subroutine DOT computes the dot, or scalar, product of two given 3-space vectors.

4.2.3 Usage.-

4.2.3.1 Calling sequence: CALL DOT (VEC1, VEC2, X)

4.2.3.2 Arguments:

Parameter name	In/Out	Dimension	Type	Description
VEC1	In	3	Real	Input vector "$\vec{A}$"
VEC2	In	3	Real	Input Vector "$\vec{B}$"
X	Out	1	Real	Scalar product

4.2.3.3 Label common: There is no label common.

4.2.3.4 Sample usage: Refer to "Calling Sequence" above.

4.2.3.5 Storage required: Coding occupies $64_8(52_{10})$ locations.

4.2.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

4.2.4 Method.-

4.2.4.1 Statement of algorithms:

$$\vec{A} \cdot \vec{B} = X$$

where $\vec{A}$ and $\vec{B}$ are given vectors and X is the magnitude

$$|\vec{A}| \quad |\vec{B}| \quad \cos \angle AB$$

4.2.4.2 Derivations or references: There are no derivations or references.

4.2.5 Restrictions.-

4.2.5.1 Range of numbers that can be processed: Arguments must be in single precision.

4.2.5.2 Range of applicability: Subroutine DOT is a general purpose single precision dot product routine.

4.2.5.3 Other programs required: No other programs are required.

4.2.6 Accuracy.- Accuracy is limited only by the hardware/software limitations of the computer.

4.2.7 Coding information.- All computations and input and output are in single precision.

4.2.8 Listing.-

```
SUBROUTINE DOT (VEC1,VEC2,X)
DIMENSION VEC1(3),VEC2(3)
X=VEC1(1)*VEC2(1)+VEC1(2)*VEC2(2)+VEC1(3)*VEC2(3)
RETURN
END
```

4.3 Subroutine CROSS

4.3.1 Identiification.-

CROSS (Vector Cross Product Routine)
F. Johnson, January 31, 1968
IBM 7094
FORTRAN IV

4.3.2 Purpose.- Subroutine CROSS computes the single precision cross (or vector), product of two given 3-space vectors. The magnitude and its square are also returned with the 3-space product.

4.3.3 Usage.-

4.3.3.1 Calling sequence: CALL CROSS (VEC1, VEC2, VEC4)

4.3.3.2 Arguments:

Parameter name	In/Out	Dimension	Type	Description
VEC1	In	3	Real	Input vector "$\vec{A}$"
VEC2	In	3	Real	Input vector "$\vec{B}$"
VEC4	Out	5	Real	The first three locations are the vector cross product ("$\vec{C}$"). VEC4(4) = Magnitude of the vector cross product VEC4(5) = The square of the magnitude

4.3.3.3 Label common: There is no label common.

4.3.3.4 Sample usage: Refer to "Calling Sequence" above.

4.3.3.5 Storage required: Coding occupies $144_8(100_{10})$ locations.

4.3.3.6 Error codes and diagnostics: There are no error codes or diagnostics.

4.3.4 Method.-

4.3.4.1 Statement of algorithms: Initially the cross product of the input 3-space vectors is computed.

$$\vec{A} \times \vec{B} = \vec{C}$$

where A and B are given vectors and C is a vector of magnitude

$$|\vec{A}| \cdot |\vec{B}| \sin \angle AB$$

and direction such as defined by the right-hand rule. Then the magnitude $|\vec{C}|$ and its square $|\vec{C}|^2$ are computed.

4.3.4.2 Derivations or references: There are no derivations or references.

4.3.5 Restrictions.-

4.3.5.1 Range of numbers that can be processed: Arguments must be single precision.

4.3.5.2 Range of applicability: Subroutine CROSS is a general purpose vector cross product routine.

4.3.5.3 Other programs required: No other programs are required.

4.3.6 Accuracy.- Accuracy is limited only by the hardware/software limitations of the computer.

4.3.7 Coding information.- All calculations and input and output are in single precision.

4.3.8 Listing.-

```
SUBROUTINE CROSS (VEC1,VEC2,VEC3)
DIMENSION VEC1(3),VEC2(3),VEC3(5)
VEC3(1)=VEC1(2)*VEC2(3)-VEC1(3)*VEC2(2)
VEC3(2)=VEC1(3)*VEC2(1)-VEC1(1)*VEC2(3)
VEC3(3)=VEC1(1)*VEC2(2)-VEC1(2)*VEC2(1)
VEC3(5)=VEC3(1)*VEC3(1)+VEC3(2)*VEC3(2)+VEC3(3)*VEC3(3)
VEC3(4)=SQRT(VEC3(5))
RETURN
END
```

4.4 Subroutine HELP

This subroutine is identical to the subroutine HELP called by CIST. Refer to section 2.9 for a detailed description.

5.0 THE USE OF SUBROUTINES CIST AND UPDATE IN PROVIDING FIRST GUESSES FOR THE RTCC PROCESSOR

The generation of first guesses for the RTCC TLI processor occurs in the following five sequential steps, which will be described in detail in the order of their occurrence.

1. Initialization before calling CIST.

2. Call CIST.

3. Initialization before calling UPDATE.

4. Call UPDATE.

5. Use of UPDATE output.

5.1 Initialization Before Calling CIST

Before CIST can be called, several things must be done.

First, the input data must be loaded into the PRE array of the SIMUL common block with proper units (section 1.3).

Second, the ephemeris must be initialized relative to a base time specified by PRE(1), PRE(2), and PRE(3). The times of all state vectors in CIST are measured relative to this base time, which is the midpoint of the 24-hour period within which the time of the input parking orbit state vector is to be defined.

Third, the right ascension of Greenwich (RAGBT) at zero hours (at base time) must be defined, in radians. RAGBT is the only variable in the calling argument of CIST.

5.2 Call CIST

Assuming successful convergence, CIST will define the following, for coplanar TLI:

1. Time (T1) of the input parking orbit state vector.

2. State vectors at the beginning and end of the TLI thrust maneuver.

3. Perigee state vector of the translunar trajectory.

4. All TLI targeting elements, including characteristic velocity.

5. Time of pericynthion.

5.3 Initialization Before Calling UPDATE

Immediately after control has returned to the program calling CIST, the convergence index COI (see TAR(1) in section 1.3) should be interrogated. Successful convergence is indicated by COI = 0.0.

If COI $\neq$ 0.0, CIST did not converge successfully and the output of CIST should be disregarded. In these unconverged cases with the present version of CIST, there are two alternatives, both of which leave much to be desired. The user can either input a different input parking orbit state vector and/or a different pericynthion position of the simulated trajectory, and then call CIST again.

Assuming successful CIST convergence, all that has to be done before calling UPDATE is to define the time dispersion (TOIDIS) of the input parking orbit state vector. As defined in section 3.1, TOIDIS is the difference in hours between the real time, actual, time of the input parking orbit state vector and that time defined by CIST as being necessary for coplanar TLI. This latter time for coplanar TLI, is denoted as T1 and is stored in location 57 in the SIMUL block, TAR(2). The sign of TOIDIS is very important.

$$\text{TOIDIS} = \begin{bmatrix} \text{real time (actual) time} \\ \text{of input parking orbit} \\ \text{state vector} \end{bmatrix} - \text{T1}$$

Both of the times in the right member of the above equation are measured relative to the G.m.t. base hour stored in PRE(3). If the G.m.t. base hour is input as the actual time of the parking orbit state vector, the first term in the above equation becomes zero, and the calculation of TOIDIS becomes greatly simplified, as indicated by the following equation:

$$\text{TOIDIS} = -\text{T1}$$

It should be pointed out that the shorter the coast time in orbit, from the input state vector to the beginning of TLI, the more accurate is the T1 defined by CIST. Consequently, the values of TOIDIS given by the above equations for input parking orbit state vectors which are approximations to the beginning of TLI, will be relatively accurate. This is because in the present version of CIST, the parking orbit is very crude, being calculated as inertially fixed and perfectly circular. Needless to say, values of TOIDIS given by the above equations for an insertion state vector will be relatively inaccurate.

5.4 Call UPDATE

As described in section 3.1, calling UPDATE will modify the position of the $\hat{M}$ TLI target vector defined in CIST for coplanar TLI, to compensate for the dispersion TOIDIS of the time of the input parking orbit state vector.

5.5 Use of UPDATE Output

The TLI targeting elements defined by CIST and UPDATE which should be used in subsequent real-time mission planning are the following:

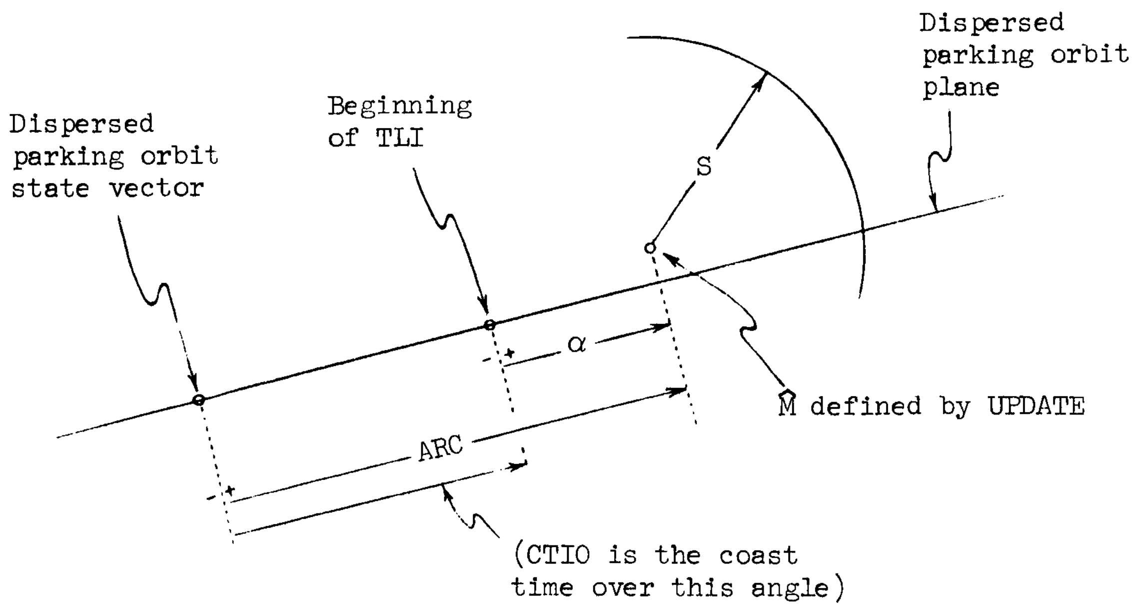
1. S, the perigee hypersurface radius [location 65 in SIMUL block, TAR(10)].

2. W, the trajectory energy at perigee [location 66 in SIMUL block, TAR(11)].

3. The updated unit $\hat{M}$ TLI target vector [Cartesian components in location 78, 79, 80 in SIMUL block, TAR(23, 24, and 25)].

It is unfortunate that S, W, and $\hat{M}$ are not the independent variables used in subsequent iterative trajectory calculations. The independent variables commonly used are S, W, CTIO (coast time in parking orbit from the dispersed orbit state vector to the beginning of TLI), and DEL (δ, the latitude of $\hat{M}$ relative to the dispersed parking orbit plane).

To begin an iteration using S, W, CTIO, and DEL as independent variables, it is first necessary to calculate values of CTIO and DEL which are compatible with the updated $\hat{M}$. To calculate this initial value of CTIO, it is necessary to know the angle α, which is the angle in the parking orbit plane from the beginning of TLI to the perpendicular projection of $\hat{M}$.



It is imperative that the angle α, used in this calculation of initial CTIO, be calculated by the same TLI simulation which will be used to calculate α and the other dependent TLI simulation variables in the subsequent iteration. If this is not done, the initial CTIO will not be compatible with the $\hat{M}$ defined by UPDATE, and the first trajectory in the iteration will not be as good a beginning as it could be.

A sufficiently accurate value of CTIO can be calculated by the following equation:

$$\mathrm{CTIO} = \frac{\mathrm{ARC} - \alpha}{\mathrm{WV}}$$

ARC is the angle in the parking orbit plane from the dispersed state vector to the perpendicular projection of $\hat{M}$ (see figure above), and WV is the angular velocity of the vehicle at the dispersed state vector. Given the update $\hat{M}$ and the dispersed parking orbit state vector, the calculation of the angles ARC and DEL are straightforward problems in trigonometry and vector analysis. These calculations of ARC and DEL are performed in UPDATE.

In addition, the hypothetical coast time (CT) in hours over the angle ARC is calculated in UPDATE.

$$CT = \frac{ARC}{WV}$$

In this calculation, the value of WV used is that of a circular orbit having the radius of the dispersed state vector. However, the wisdom of calculating CT in UPDATE is questionable, for invariably someone confuses CT with CTIO, regardless of how plainly CT is defined in the UPDATE printout.

REFERENCES

1. Johnson, Francis, Jr.: The Calculation of Trajectories by the Empirical Simulation of Restricted State Vectors. MSC IN 68-FM-21, January 23, 1968.

2. Johnson, Francis, Jr.: The Impulsive Simulation of Optimum TLI Thrust Maneuvers. MSC IN 67-FM-125, August 28, 1967.

3. Johnson, Francis, Jr.: Translunar Injection Tangency Surfaces - An Empirical Mechanism for Predicting Free-Return Trajectory Planes Near Outgoing Perigee. MSC IN 66-FM-10, February 18, 1966.

4. Danby, J.: The Fundamentals of Celestial Mechanics. Second printing, 1964. The Macmillan Co. New York.

5. Better TLI Simulation Equations. MSC Memorandum 67-FM56-190, June 5, 1967.

6. Complete Set of Empirical Equations for the "Black Box" Simulation of Optimum Coplanar TLI's from Circular Orbit. MSC Memorandum 67-FM56-220, June 19, 1967.

7. Johnson, Francis, Jr.: Modifications of TLI Targeting First-Guess Logic for Dispersions in Launch Time. MSC Memorandum 66-FM52-241, July 27, 1966.

8. Johnson, Francis, Jr.: Checkout of Method of Updating TLI Targeting Elements for Dispersions in Launch Time. MSC Memorandum 66-FM56-351, October 12, 1966.